Ján Janikovič

Gas Turbine Transient Performance Modeling for Engine Flight Path Cycle Analysis

School of Engineering
Gas Turbine Engineering Group

PhD Thesis

Supervisors: Prof. P. Pilidis, Prof. R. Singh

January 2010

CRANFIELD UNIVERSITY
School of Engineering
Gas Turbine Engineering Group

PhD Thesis

Ján Janikovič

Gas Turbine Transient Performance Modeling for Engine Flight Path Cycle Analysis

Supervisors: Prof. P. Pilidis, Prof. R. Singh
January 2010

This thesis is submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

ABSTRACT

The growth in competitiveness in airline industry has called for more advanced tool to estimate the operating costs. Engine maintenance costs are an important decision-making element during airline fleet selection judgment. Long term observation in aerospace led to the development of engine maintenance costs calculators based on empirical correlation. But the possibilities of empirical model application for future engines without prior operational data are limited.

A physics-based tool to estimate the life of the engine components and predict the shop visit rate requires the variations of thermodynamic parameters over the flight path. High fidelity engine models are simulated using an engine performance program.

A test program designated for design, off-design and transient performance simulation for simple turbojet layout gas turbine engine has been programmed and tested. The knowledge gained from program coding was used to generate more robust transient performance code implemented to Turbomatch. Two transient methods have been tested: The rapid transient performance method and the thermodynamic matching method. The tests showed greater robustness and stability of the second method, which has been finally adopted for the program. For industrial engine configuration and for future novel engine cycles the heat-exchanger dynamic response model was implemented and tested.

Created tool was demonstrated on short-haul study of engine flight path analysis. Together with the aircraft model, the tool produced variations of parameters needed for the lifing algorithm.

ACKNOWLEDGEMENTS

The overall outcomes of any research are deliverables of not only a single researcher, but of a whole group of people, sharing their thoughts, giving the guidance and providing the inspiration and motivation during the whole project.

I would like to express my greatest gratitude to Professor Pericles Pilidis, Professor Riti Singh and Dr. Peng Ho for their guidance, advices and tremendous support. Their thoughts and ideas had the major influence on the project. I am very grateful to the Boeing Company to take place in funding the project. I would like to thank to Dr. Hariharan Hanumanthan, who has been my colleague through the whole time of the project. Apart from his infinite friendship he did not hesitate to share his experiences and wisdom. An appreciation goes to Dr. Yiguang Li and Richard Hales especially for useful comments (positive and critic) on Turbomatch development.

Thanks goes to Heather Woodfield my mother Iveta Janikovicova who took the time in checking some parts of the thesis and shared their ideas on its improvements. Thanks to Valerio Lallini, Luca Saltelli, Francesc A. Pastor and Ausias Pelicer for their effort in the subproject. I am thankful to Gillian Hargreaves, Sam Broe, Nicola Datt, Maria Negus and Claire Bellis for their regular assistance.

I would like to thank to my family who gave me the essential psychical support in the difficult time of the research. A special thanks goes to my dear Adela Kremzova.

My sincere thoughts go to Navaal Ramdin from the Boeing Company. His friendship and infectious good mood has been spread around until he left this world in a very early age.

LIST OF CONTENTS

NOMENCLATURE

Symbol	Definition	Unit
A	Contact surface area	$[m^2]$
a_m	Corrected mean acceleration	$[m.s^{-2}]$
COT	Combustor outlet temperature	[K]
c_p	Specific heat capacity at constant pressure	$[J.kg^{-1}.K^{-1}]$
CR	Smaller ratio of fluid capacities	[-]
c_v	Specific heat capacity at constant volume	$[J.kg^{-1}.K^{-1}]$
d or Δ	change in value (increment or decrement)	
f	fluid friction coefficient	[-]
F_0	Net thrust at M = 0	[N]
F_{LOF}	Net thrust at lift-off point	[N]
F_m	Corrected mean net thrust	[N]
h	Specific enthalpy	$[J.kg^{-1}]$
H	Enthalpy	[J]
H	Passage height	[m]
h	Heat transfer coefficient	$[W/(m^2K)]$
I	Rotor inertia	$[kg.m^2]$
k	Fluid thermal conductivity	$[W \cdot K^{-1} \cdot m^{-1}]$
L	Heat exchanger length	[m]
M	Mach number	[-]
m	Mass	[kg]
N	Rotational speed	[Hz]
Nu	Nusselt number	[-]
p	Pressure	[Pa]
PR	Pressure ratio	[-]
Pr	Prandtl number	[-]
R	Specific gas constant	$[J.kg^{-1}.K^{-1}]$
r	Universal gas constant	$[J.K^{-1}.mol^{-1}]$
Re	Reynolds number	[-]
s	Entropy (or wall thickness)	$[J.K^{-1}]$ (m)
T	Temperature	[K]
t	Time	[s]
t_{GR}	Ground roll duration	[s]
U	Overall heat transfer coefficient	$[W/(m^2K)]$
v	Volume	$[m^3]$
V	Velocity	$[m.s^{-1}]$
V_{EF}	Effective velocity, at which the engine failure occurs	$[m.s^{-S}]$
W	Mass flow	$[kg.s^{-1}]$
Y	Contact area widht	[m]
Z	Compressor surge margin	[-]

Greek symbol	Definition	Unit
γ	Heat capacity ratio	
δ	Inlet pressure / 101 325 Pa	
ε	Tolerance	
η	Efficiency	
θ	Inlet temperature / 288.15 K	
λ	Proportion of extracted fluid	
μ	Dynamic viscosity	$[kg \cdot m^{-1} \cdot s^{-1}]$
ν	Kinematic viscosity	$[m^2 \cdot s^{-1}]$
π_C	Compressor pressure ratio	
ρ	Density	$[kg/m^3]$
Φ	Isentropic function	
χ	Convection increase factor	
Φ_H	Hydraulic diameter	[m]

Acronyms and subscripts	
a	Air
A	Component inlet
B	Combustion chamber (Burner)
B	Component outlet
B1	Inlet to component control volum
C	Compressor
cold	Colder stream
COT	Combustor outlet temperature
CP	Compressor power
CPU	Central processing unit (computer microprocessor)
cr	Critical
DMC	Direct maintenance costs
DP	Design point
EFPCA	Engine flight path cycle analysis
FAR	Fuel-to-air ratio
ff	Fuel flow
FPA	Flight path analysis
g	Combustion gasses
GR	Ground roll
GT	Gas turbine
GUI	Graphical user interface
hot	Hotter stream
HPC	High-pressure compressor
HX	Heat exchanger
IGV	Inlet guide vanes
is	Isentropic
ISA	International standard atmosphere

j	Jet
LOF	Lift off point
NGV	Nozzle guide vanes
NLE	Non-linear equation(s)
NTU	Number of transfer unites
OD	Off-design
OOP	Object-oriented programming
PCN	Relative rotational speed
PR	Pressure ratio
rd	referred variable
s	Static conditions
SDK	Software development kit
SLS	Static level sea
SP	Surplus power
t	Total (stagnation conditions)
T	Turbine
T-O	Take-off
TP	Turbine power
TR	Transient
TSFC	Thrust specific fuel consumption
vol	Volume

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The increase in competitiveness and expansion in airline and aircraft manufacturing industry have called for more advanced cost evaluating methods. Airline expenses can be sub-divided into groups from which the maintenance and partly fuel costs are expenses that the airline manufacturers can influence either by aircraft design or by engine selection. Fuel costs can be estimated relatively accurately by available flight path analyzing software like Hermes or FLOPS using aircraft size and weight data and engine SFC data. According to the analysis the aircraft manufacturer can offer the customer an aircraft fitted with engine with better fuel economy. Airline companies then can further eliminate expenses by buying fuel on cheaper airports and if the fuel price distinction between two airports of an air route is too high the airline can decide to fill the tank with fuel for both onward and return journey. This, however, results in higher aircraft weight and consequently higher thrust must be used for take-off, climb and cruise, which reduces the engine components life and increases engine maintenance costs and shop visit rate.

Further life extension of engine parts can be achieved by applying the derate, that means lower maximum thrust than it is available for the engine, when ambient conditions are suitable and runway length is long enough. Applying derate is sensible only when it is possible to evaluate the impact on life consumption.

For direct maintenance cost evaluation a lifing model was developed to estimate the life consumption dependencies on derates or ambient and take-off condition during flight. Prior to life estimation calculation and engine flight path cycle analysis (FPA) is

carried out which will give the variation of thermodynamic parameters during the flight that are important for life estimation. Engine flight path cycle analysis is divided further into two groups: Engine performance simulation and Flight path analysis. Within the scope of Engine performance simulation an engine model is created and simulated to obtain engine performance data for Flight path analysis. From the task determination the requirements for the engine model result: The engine model must be simulated with fidelity. All gas paths affecting the component life must be tracked. The computational tool working with the engine model must be flexible and robust to deliver the data for any possible engine configuration.

Cranfield University program, Turbomatch, has been chosen for this purpose. The program has been developed and tested for several decades to ensure it will able to simulate engine behavior under any operating steady state conditions. Steady state simulation is perfectly satisfactory if operating conditions are stationary. But when the operating conditions start to vary significantly (during take-off, reverse thrust, any change in throttle settings...) the engine response is different as compared to the steady state. An algorithm has been adopted and tested to model these scenarios.

After the engine model has been fixed, an aircraft model is created and its flight cycle simulated across selected flight paths. The choice of selected flight paths determines usually current aircraft air routes or air routes planned for the aircraft in the future. The whole flight path may be either simulated under steady state conditions with the loss in accuracy. A method is proposed to simulate the take-off segment under transient conditions.

In order to be able to model future engine configuration possibly involving intercooled-recuperated cycle under transient performance conditions, the heat exchanger model has been implemented to the engine modeling tool.

CHAPTER 2

THE DEVELOPMENT OF TURBOMATCH CODE

2.1 Turbomatch - Historical Background

The first computer programs designed to simulate the performance of any kind of gas turbine engine cycles began to appear in the 1960s. Until then, few programs to simulate specific engines were available, and most of the calculations of were carried out manually. With the increasing use of computers the demand grew for more flexible engine cycle simulation programs able to model different types of engines. These programs could not have been developed previously, not only because computers were very expensive, but also because the available programming languages did not allow that level of control for either the programmer or the program user. Programming languages such as Algol and FORTRAN 66 enabled the creation of a structured program, and as such they were used for the development of engine cycle calculation codes. The earliest programs were designed to calculate only the design point of engines that followed certain component configurations (Chappell, et al. 1964; Cockshutt, et al. 1965). In 1967, the Turbocode program was developed, with an approach whereby particular engine components were built up together in the input file as required, which enabled the user to simulate the design point of any engine configuration (Palmer 1967; Palmer, Annand 1968). That year also saw the development of the first design simulation programs. The NASA program SMOTE employs modified Newton-Raphson iteration and component characteristics (maps) are used to calculate quantities across components. These maps are adjusted to particular engine design by applying scaling factors (McKinney 1967). The limitations of SMOTE's flexibility led to the development of GENENG and GENENG II (Koenig, Fishbach 1972). Nevertheless, the flexibility of even the GENENG programs was limited to standard engine configurations.

The original aim of the development of Turbomatch carried out at Cranfield University was to create a program that would be able to simulate in design and off design analysis of any kind engine configuration, regardless of complexity. The first version of Turbomatch appeared in 1974 and enabled simulations of design point and off design of any kind of engine configuration (MacMillan 1974). It was written in FORTRAN 66, the same programming language that was used for the GENENG programs. Since then, Turbomatch has become the fundamental simulation tool for gas turbine cycle and performance research projects at Cranfield University. The development of Turbomatch continued and further improvements have been made, the most significant being:

- 1991 The addition of the convergent-divergent nozzle, the intercooler brick and the help menu (Van den Haut 1991). Turbomatch developers also started to use FORTRAN 77 features.
- 1992 The addition of variable geometry simulation for compressor IGVs and turbine NGVs. The program has been made independent of VAX/VMS and could be run on a PC (Siringlou 1992).
- 1994 – 1995 User-defined scaling factors during off design – improvements to engine degradation study (Patrick Escher, Louay Aleid).
- 1998 – 2007 The period of most significant reconstruction of the program code (Richard Hales). The program was rewritten from FORTRAN 77 to Fortran 90 standards (Ellis 1994). This was done for three main reasons. The original Turbomatch code written in FORTRAN 77 used many GOTO loops in places where the combination of using IF blocks and calling subroutines would have been a better option. This resulted in a so-called “spaghetti program” structure, which is hard to work with, especially on an extensive program such as Turbomatch, with 13 032 lines (including comments). The program has been restructured to enable faster and easier future development by students at Cranfield University. Although there were 36 057 lines of code after the reconstruction, some purpose-related functions and subroutines have been put together into modules and files. The second reason

was to rewrite the program to conform to the new standards of Fortran. Until Fortran 95 became standardized all programs written in FORTRAN 77 were fully compatible with Fortran 90 compilers. This compatibility of FORTRAN 77 however is no longer valid with Fortran 95 and later standards. If the code remained unchanged it might be not possible for future Fortran compilers to run the program. The third reason was to prepare the code for operation under the Windows NT platform, which was done by adding a few features of DIGITAL Visual Fortran, followed by the development of a graphical user interface (GUI) – Pythia – which carried out pre-processing and post-processing activities (Yiguang Li). Turbomatch became the computing core (the processor) integrated into Pythia. Currently it is even possible to control engine parameters during calculation in Pythia, for example when diagnostics and deterioration are simulated. Since Visual Fortran's GUI abilities are much more limited than those of other languages intended for GUI programming (e.g. Visual C++, Visual Basic, Java, etc.), Pythia became a mixed-language program, using Visual Basic as the programming language for GUI and Fortran 90 for Turbomatch routines for engine cycle calculation.

2.2 Description of the Original Code

As there have been many changes made at different times throughout the existence of Turbomatch, there are many versions of it. Only three versions of the program will be mentioned here:

Turbomatch Legacy

This version of Turbomatch can be accessed by the students and staff of Cranfield University on external U: drive by logging into the University network. Because it has been widely used in most gas turbine performance simulation projects for more than a decade, its results and abilities have been tested thoroughly. It is possible to build a model of any kind of existing gas turbine regardless of its complexity. The last major adaptation was carried out by Patrick Escher in 1995, and minor amendments and corrections and debugging could have been implemented until 6 September 2006

(chapter 2.1). Code is written in FORTRAN 77. It is stable and it can simulate both design and off design of gas turbine engines. Program code consists of 13 032 lines (including comments) located in 25 files.

Turbomatch Intermediate

The work on this newer version was initiated by Mr. Richard Hales at the end of 1998 and has been continued by him and Dr. Yiguang Li, greatly improving the accuracy of the code and degradation simulations. The objection was to prepare the code for Windows NT with DIGITAL Visual Fortran in MS Development Studio. Program code was rewritten into Fortran 90 accompanied by major changes in the structure that were necessary for future development. For example, the program was split into more files where every file contained only one module, subroutine or function. In the future it will be possible to create external objects from these files that can be amended independently of the code. This step has therefore extended the size of the code. In 2007 Marco Mucini implemented humidity and multifuel capabilities into the program. Gas properties routines have also been changed, such that consequently the engine cycle simulation results of the Intermediate version of Turbomatch no longer match the results of the equivalent engine model run on Turbomatch Legacy version.
For testing purposes however, many Legacy code snippets have been preserved in the Intermediate version, resulting in following issues:

1. The code sometimes performed the same operation twice as some shadow routines were left in the code. A shadow routine is generally an unchanged (obsolete) subroutine or function copied from the Legacy version which does the same thing as its substitutive subroutine written in modern computer language structure. Routines from the Legacy version normally work with shadow (obsolete) variables, whereas new routines work with new variables. Using multiple routines and variables allow the code developer to see whether new routines return the same results as their shadow versions. For example, subroutine newDATIN, which is responsible for loading brick data into specified

variables, has its shadow subroutine DATIN, which resembles the structure of the same subroutine in the Legacy version, is responsible for the same task. The same applies for CUNPAC, CODIN and ARITHY.

2. The same variable was often stored twice (into two different variables – the new and shadow variable). If the first and second brick data of the intake corresponded to the altitude and the ambient temperature deviation from ISA, their values were stored in both, BD(1) and BD(2) arrays and own data types bricktype1(1)%Altitude and bricktype(1)%ISADev, where the first way of storing brick data came from the Legacy version and the second represents the updated way. Likewise for station vector variables, where SV(i) array emanated from the Legacy version and Station(i) own data type represents the new way for storing engine station vectors. Although it was intended that the updated version of Turbomatch would only use bricktype*(i) and Station(i) data types for operations, and obsolete variables would only serve the purpose of establishing whether calculations in subroutines return the same values, not all subroutines were rewritten to work with data types variables. Most calculations were either carried out using both Legacy and Intermediate versions variables (alternatively calculation were done with one type of variables and the result was then copied into another), but sometimes the calculation was carried out only in one version variable. This resulted in a necessity for the programmer to check if variables have up-to-date values any time change in the code was added.
3. There were two types of comments that made the readability of the code much harder:
    - Out-of-date comments (e.g. they analyzed a problem of the code that already has been solved)
    - Commands and snippets of the code that were “turned off” by putting it into a comment by exclamation mark “!” in front of the command line.

Turbomatch TR (Transient)

The development started in the beginning of 2007 and it was concluded on 20 July 2009. Originally there were two main improvements to develop:

- To implement transient performance simulation capability
- To develop and program heat exchanger models for engine transient performance analysis

TR version emanates from unfinished Intermediate version. Both improvements have been added to a code with consequential benefits that resulted from the development. Since the transient performance simulation is expected to be used especially for flight path analyses the control of fuel schedule has been implemented. Also convergence abilities and code stability was improved. Obsolete code snippets, commands and comments were removed what improved the code readability. Appendix 2.1 also demonstrates how the source code clarity improved on INTAKE subroutine calling sequence. In the thesis the development of Turbomatch TR is described.

2.3 Turbomatch Code Clean-up

The received code of Turbomatch version initially consisted of 36 057 lines including comments and deactivated code snippets. Because of the earlier-mentioned development consequences (chapter 2.2) the code was difficult to work with and it also took longer time to understand the program code. Before any changes were applied to the code it needed to undergo a cleaning-up process. It is worth noting that several attempts were made to work even with the original code. The programming went relatively fast but the debugging was slowed down mainly because it was difficult to determine what the cause of any error that occurred was.

As stated in (Ellis 1994) errors that occur in programs are divided into:

- syntactic (grammatical) errors
- semantic (logical) errors

Syntactic errors are not of a great issue as they are detected by the programming language compiler. On the other hand, **semantic** errors result in incorrect program logic. They can cause unexpected termination of a program during execution or that a program returns wrong results. This kind of programming errors is far more serious since such errors cannot be simply detected just by computer. The process of manual error detection is called debugging and it is mostly done by running the code in debugging mode where the programmer can stop the runtime at any part of the code and check the values that are loaded in variables at that point of time. It was debugging what consumed most of the time in the code development. And in most cases the problem was not to run the code, but to achieve the convergence of the engine model in off-design simulation – a problem obviously caused by semantic errors. Following text describes what changes were applied to the code in the clean-up process:

2.3.1 Irrelevant Comments Removal

The part of the code that is not processed by the compiler is called a comment. In Fortran it starts with an exclamation mark "!" and finishes at the last character on the same line. A comment is included only for information to person who reads the code. It can be used to describe the task of related code snippet, to provide information on declared variables, to explain the algorithm that is used in the program, to describe the routine interface, etc. Many comments were inherited from developing days that were not relevant for the code anymore. Although they did not cause any problem to the logic of the program those comments, that made readability of the code worse, were removed without changing the program logic (appendix 2.1). The comments included:

- *Programmers' communication:* In developing stages of Turbomatch comments were also used as a means of communication between code developers or as a means of remainder for problem that was postponed to the future. These comments were not very excessive throughout the code and many of them were left in the code if they did not made the readability harder.

- *Turned off code snippets:* Lines of the code that were not needed in updated version of program. They often emanated from the Legacy version of Turbomatch. These types of comments were often the cause of hard readability of the code.

2.3.2 Shadow Routines Removal

DATIN, CUNPAC, CODIN and ARITHY were routines that were removed as their tasks were replaced by other routines with similar names.

2.3.3 Shadow Variables Removal

These included for example arrays BD, SV, ER, VAR, etc. Similarly as for shadow routines these also had their substitute (eq. for BD it was bricktype* own datatype). Shadow routines and shadow variables removal had significantly reduced calculation time which was particularly slowed down during off-design iterations before.

2.3.4 Files List Sequence Change

In the Solution Explorer, which provides the full list of code files used for program compilation, the order of files was somewhat impractical. Generally, every file contained only one subroutine or one module so when one wanted to look for a routine with specific name he just needed to search for the file whose name contained the name of the subroutine. Original file names started normally with word, following by a number so the sequence of routines did not follow alphabetical order of routines names. Changing their names in a way that sorts them alphabetically allows faster searching for any routine (appendix 2.2)

2.3.5 Portability and Compatibility

Due to intrinsic limitations of a computer, program variables declared as REAL store only an approximation of the original number. The degree of precision of the variable depends on number of digits that CPU assigns to variable. In default 32-bit computer environment intrinsic REAL variables have 6 digit precision and the exponent ranges from -10^{38} to $+10^{38}$. If Computer carries out an arithmetic operation between two numbers from which one is by few orders higher than the other, the result accuracy

decreases significantly (appendix 2.3). Indirect evidence proving that precision affects convergence abilities significantly was an experiment where the same engine was run in off-design simulation mode once with default precision set REAL declared variables and once with increased precision. The simulation did not converge when default precision was used but when the precision was increased the simulation converged without any difficulties. The default precision in 64-bit computer environment is higher than in 32 bit environment and it would most probably suffice for successful engine cycle simulation.

To ensure sufficient precision at any computer a module called **PortSet** was created. In this module variable RK was declared which is assigned to have value of REAL kind that returns a precision of at least 10 valid digits and exponential range 10^{-60} to 10^{60} when variable is declared with this kind. Then all REAL variables in Turbomatch are declared with RK kind (More information about variables kind can be found in Ellis (1994) book, in chapter 10)

Alternative approach would declare REAL variables as DOUBLE PRECISION. Thus floating-point variables would have sufficient precision in default 32-bit environment. However, in 64-bit environment these variables would have much higher precision than needed, which would result in higher proportion of RAM taken by the program. Moreover it would unnecessarily increase the computational time.

2.3.6 NLE System Solver Update

Subroutine ErrProc is the main equation solver for Turbomatch non-linear equations. It uses Newton – Raphson method to handle guesses for variables and processes errors to produce variables (Zhidkov, et al. 1965). An ability to cope with variables whose initial value equals to zero has been added.

The essence of the procedure is to find the roots for the set of non-linear equations:

$$f_i(x_1, x_2, ..., x_n) = 0 \qquad \text{E2.3.1}$$

For example the engine mass flow compatibility equations E.2.4.22 and E2.4.23 are transposed into non-linear equations of E2.3.1 form. In this example the actual mass

flow entering and leaving an engine component represents the unknown (variable) x_1 and the compressor pressure ratio represents the unknown x_2 in E2.3.1. A matrix containing the differentials of these equations is created. This Matrix is also referred to as Jacobian:

$$J_{ij} = \frac{\partial f_i}{\partial x_j} \Rightarrow J = \begin{pmatrix} \frac{df_1}{dx_1} & \frac{df_1}{dx_2} & \cdot & \frac{df_1}{dx_n} \\ \frac{df_2}{dx_1} & \frac{df_2}{dx_2} & \cdot & \frac{df_2}{dx_n} \\ \cdot & \cdot & \cdot & \cdot \\ \frac{df_n}{dx_1} & \frac{df_n}{dx_2} & \cdot & \frac{df_n}{dx_n} \end{pmatrix} \qquad \text{E2.3.2}$$

Particular derivatives can be calculated by:

$$\frac{df_i}{dx_j} = \frac{f_i(x_j + \varepsilon) - f_i(x_j)}{\varepsilon},$$

where ε is the required error bound.

Using the Tailor series:

$$f_i(\vec{x} + d\vec{x}) = f_i(\vec{x}) + \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} dx_j + \varphi(d\vec{x}^2) \qquad \text{E2.3.3}$$

and neglecting increments of the order d$\mathbf{x}^2$ and higher, and setting $\vec{f}(\vec{x} + d\vec{x})$ to be equal to zero (see E2.3.1) a new system of linear equation is obtained:

$$\vec{J} \cdot d\vec{x} = -\vec{f} \qquad \text{E2.3.4}$$

This system is solved by LU Decomposition method explained later. After the solution vector of this set of equation was found, the new value of the parameter is determined by:

$\mathbf{x}_{i+1} = \mathbf{x}_i + d\mathbf{x}$

This iteration is repeated again and again, until one of the conditions:

1. $\vec{f}(\vec{x}_{found}) < \varepsilon$
2. $\vec{x}_i - \vec{x}_{i-1} < \varepsilon$ where i stands for the number of iteration

is fulfilled

LU Decomposition

There are more suitable methods of how to solve a system of linear equations E2.3.3.The principle of this method is a matrix, decomposed into two triangular matrixes (Press, Teukolsky 1994):

L . U = A E2.3.5

where **A** is the original matrix, **L** is the lower triangular with α_{ii} elements and **U** is the upper triangular matrix β_{ii} elements.

Then the set of equations: **A . x = b** can be decomposed to **L . (U . x) = b**

So it is possible to solve initially :

L . y = b E2.3.6

using:

$$y_1 = \frac{b_1}{\alpha_{11}}$$

$$y_i = \frac{1}{\alpha_{ii}}\left[b_i - \sum_{j=1}^{i-1} \alpha_{ij} y_j \right] \qquad i = 2,3,..., n$$

and then:

U . x = y E2.3.7

using

$$x_n = \frac{y_n}{\beta_{nn}}$$

$$x_i = \frac{1}{\beta_{ii}}\left[y_i - \sum_{j=i+1}^{n} \beta_{ij} x_j \right] \qquad i = n-1,n-2,...,1$$

For **L** and **U** matrix generation a Crout's algorithm has been used, which considers α elements on the diagonal to be equal 1. Further details about the Crout's algorithm can be found in (Press, et al. 1994).

The procedure requires a first guess which should be close to the root. The original solver for system of linear equations (containing Jacobian) was replaced by LU matrix solver because it is claimed to find solutions faster compared to other linear equation solving methods (Press, Teukolsky 1994).

2.3.7 Tolerances of Permissible Errors in Internal Iteration Loops of the Routines

Internal iteration loops are used for variables which depend on parameters that are also function of the searched variable (e.g. static temperature calculation). To find the solution the difference between guessed and calculated value of the variable must lie in user-specified tolerance. If it doesn't, calculated value of the variable is set as a new guess. If the tolerance is set to higher value, simulation fails to meet the main program convergence tolerance stored in TOLNCE. Tolerances of internal iterations were therefore lowered.

2.3.8 General Minor Corrections

The last corrections mentioned here were mainly introduced to improve the overall source code appearance and to conform to modern Fortran standards:

- All remaining COMMON block were replaced by modules
- All keywords in lower case according to Fortran 95/2003 standard (Metcalf, Reid & Cohen 2004)
- The for007, which is secondary unit for engine model input file, has been removed
- Standard source code format applied to all source code files

2.3.9 The Initial Debugging

Previous sub-chapters of chapter 2.3 explained the details about source code clean-up that was necessary to make the Turbomatch source code more synoptic. The changes mainly influenced the most important routines of the code that are responsible for engine model brick sequence order and procedures responsible for component matching iteration control. For example, the number of lines in subroutine TURBOMAT() dropped from 3615 to 1726. The final number of lines does not only represent the original source code lines, but it also includes the lines added during Transient performance implementation and Heat exchanger transient algorithm implementation. TURBOMAT() is the main subroutine of the program. It acts as a processor because it controls the calling sequence of other subroutines. On the other hand, many subroutines that did not require any modifications did not need to be involved in the cleaning process at all.

The source code clean up is especially beneficial (or even inevitable) for future Turbomatch developers who will now need to spend significantly less time to learn how the code works and implement any new algorithms into its source code. Changes in chapters 2.3.5 and 2.3.7 also improved Turbomatch convergence abilities, however the problem with unexpected program crashes remained almost unchanged as compared to Turbomatch Intermediate version. Virtually, the program only worked with certain engines model and even those were restricted to very small steps in change of parameters. Simulations of complex engine models crashed sometimes even without delivering the results for design point. The problem was analyzed and fixed before transient algorithm implementation commenced. A set of 11 engine models has been created with different configuration:

- One-spool configuration engine models:
    - Basic turbojet
    - Turbojet with combustion chamber diffuser
    - Turbojet with convergent-divergent exhaust nozzle

    - Gas generator with power turbine
- Two-spool configuration engine models:
    - Basic two-spool configuration without bypassing flow
    - Engine model of CFM56-7B interpretation
    - Engine model of GE90-90B interpretation
    - Engine model of GE 90-85B interpretation
    - Two-spool gas turbine engine model with intercooled-recuperated cycle
- Three-spool configuration engine models:
    - Three-spool turbofan with separated exhausts
    - Engine model of RR RB211 interpretation with mixed exhaust

All engines have been run by Turbomatch in debugging mode until all causes of program crashes have been located and fixed.

2.4 Engine Cycle Testing Program

Before any changes were applied into Turbomatch an experimental engine cycle simulation program Simgaen (SIMulator of GAs turbine ENgines) has been created which carries out basic gas turbine simulations based on the same (or similar) methods as iterations in Turbomatch. Since Turbomatch code was after years of development very extensive and complex, any change that was needed to be applied into its algorithm consumed a great amount of programming and debugging time. Moreover if the program failed to converge to a solution the cause of occurred failure was often ambiguous. It could be a wrong input data file (or badly designed engine model), a mistake in developed simulation algorithm or a programming bug (caused for example by mistyping). Simgaen is smaller and much simpler program compared to Turbomatch consisting only of 3590 lines. Although Simgaen enables to simulate only one engine configuration, because of its simplicity it enables faster debugging and hence faster algorithm testing. Two main transient performance methods have been tested each of them with small modifications that were necessary for using them in Simgaen, and consequently in Turbomatch. A brief discussion on thermodynamics and component

modeling used in Simgaen and comparison with methods used in Turbomatch will be done in this section.

2.4.1 Thermodynamics

To calculate the effect of compression and expansion on temperature, pressure and mass-flow changes thermodynamic calculations derived from equation of state (E2.4.1 – E2.4.3) and second law of thermodynamics (E2.4.5) are used. From all equations of state the equation of state in derivative form (E2.4.1) suits for further formula transposition most.

The equation of state:

$$RdT = p\,dv + v\,dp \qquad \text{E2.4.1}$$

The equation of state for a perfect gas:

$$pv = RT \qquad \text{E2.4.2}$$

Van der Waals equation of state:

$$\left(p + \frac{a}{v^2}\right)(v - b) = RT \qquad \text{E2.4.3}$$

Specific gas constant:

$$R = c_p - c_v \qquad \text{E2.4.4}$$

The Second Law of Thermodynamics:

$$ds = \frac{dQ}{T} \Rightarrow T\,ds = du + p\,dv = dh - v\,dp \qquad \text{E2.4.5}$$

$$ds = \frac{c_p}{T}dT - \frac{R}{p}dp \qquad \text{E2.4.6}$$

Compression occurs in intake (ram compression) and compressor, and expansion in turbine and nozzle. E2.4.8 describes the isentropic process of compression and expansion, thus no losses are taken into account. Φ_i is an entropy function which changes with temperature. Knowing the pressure or temperature difference of the process allows calculating thermodynamic gas parameters by an iterative process. Using equations E2.4.6 and E2.4.9, it is then possible to evaluate pressure or

temperature difference for a non-ideal process, given that the isentropic efficiency is known.

Specific heat capacity at constant pressure:

$$c_p = \left(\frac{\partial h}{\partial T}\right)_p \qquad \text{E2.4.7}$$

For isentropic processes:

$$ds = 0 \Rightarrow \int_{T_1}^{T_2} \frac{c_p}{T} dT = R \ln \frac{p_2}{p_1} = \phi_2(T) - \phi_1(T) \qquad \text{E2.4.8}$$

Isentropic efficiency:

$$\eta_{is} = \frac{dh_{is}(T)}{dh(T)} \qquad \text{E2.4.9}$$

Although it is possible to calculate thermodynamic parameters without iteration assuming constant working gas properties and decrease the computational time significantly, such an approach could not be used in the program as it would introduce inaccuracies from real process as high as 25% (Walsh, Philip 2004). Thermodynamic parameters calculations in Simgaen and Turbomatch use variable working fluid properties (appendix 2.4) offering higher degree of accuracy.

For combustion process simulation the interpolation of a chart of temperature rise vs. FAR has been used (appendix 2.5). Such chart is obtained from past experiments.

2.4.2 Component Maps

Engine component performance is described by a group of non-dimensional parameters. These parameters were introduced because every engine parameter depends on others and it would be very time-consuming and impractical to obtain all dependencies on testing facility and impractical to store all these data. Moreover, on a testing facility many of the operating conditions are impossible to reach (Mattingly 2006). Non-dimensional parameters can be obtained for example by Buckingham Pi Theorem analysis and they will be formed of p_t, T_t, W and N (Saravanamuttoo 2001). For component performance description four non-dimensional parameters are used:

$$\frac{p_{t2}}{p_{t1}}, \frac{T_{t2}}{T_{t1}}, \frac{W \times \sqrt{RT_{t1}}}{\sqrt{\gamma} D^2 p_{t1}}, \frac{ND}{\sqrt{\gamma R T_{t1}}}$$

where:

$\frac{p_{t2}}{p_{t1}}$ - Pressure ratio between stations 1 and 2

$\frac{T_{t2}}{T_{t1}}$ - Temperature ratio between stations 1 and 2

$\frac{ND}{N_{DP} \cdot \sqrt{\gamma R T_{t1}}}$ - Relative non-dimensional rotational speed

$\frac{W_1 \times \sqrt{R_1 T_{t1}}}{\sqrt{\gamma_1} D_1^2 p_{t1}}$ - Non-dimensional mass flow

In their corrected form quantities γ, D and R are omitted and temperature and pressure are substituted with ratios θ and δ:

$$CN = \frac{N}{\theta_1 \cdot N_{DP}} \qquad \text{E2.4.10}$$

$$CMF = \frac{W_1 \times \sqrt{\theta_1}}{\delta_1} \qquad \text{E2.4.11}$$

where:

$$\theta_1 = \frac{T_t}{T_{ISA\,SLS}} \text{ and } \delta = \frac{p_t}{p_{ISA\,SLS}} \qquad \text{E2.4.12}$$

$T_{ISA\,SLS} = 288.15\ K$

$p_{ISA\,SLS} = 101325\ Pa$

The chart showing the relation between these parameters is called component characteristic, or component map. This map is constant for a component once its geometry has been fixed. During the preliminary design of a gas turbine when the compressor or turbine maps are not yet available a comparable map must be used. Appropriate compressor map can be extracted from an existing library and scaled in order to match new compressor design point concept either by a conventional scaling technique where two points of the map are compared and respectively scaled or other

advanced scaling techniques may be used (Kurzke, Riegler 2000). When the test rig data are available for a limited range of speeds the rest of the map can be interpolated (Kurzke 1996). Alternatively it may be generated using algorithms (Kong, Kho & Ki 2004). Simgaen uses compressor maps generated by the latter approach. It plots the compressor pressure ratio and isentropic efficiency against inlet mass flow and corrected rotational speed. The map is linearly interpolated to match design point (appendix 2.6). Contours of the map are very smooth, plotted on twenty speed lines. Not so smooth and detailed maps were used for turbine map. Simgaen uses Turbomatch turbine map number 5 which is less detailed then compressor map (appendix 2.6).

Turbomatch component maps

Compressor and turbine maps used in Turbomatch legacy version were extracted from GENENG II (Fishbach, Koenig 1972) and SMOTE program (McKinney 1967) adapted only to conform with different units. They are stored in two common blocks: TMCOMMAP.FI with compressor map and TMTURMAP.FI with turbine map. There are five compressor maps using 10 speed lines (or contours) each of them consisting of 5 points (appendix 2.7), which is incidentally less smooth then Simgaen compressor map where each contour consists of 20 points. Nevertheless, the maps are smooth enough for the program to run successful engine simulation. Six maps are used to describe a turbine performance. Each of them has 10 speed lines generated from six points (appendix 2.8). Bigger curvature of speed lines causes that the map looks rougher than compressor map. Smoother turbine maps and the use of more speed lines could improve program convergence capabilities. The turbine map number one is especially problematic as it has extremities on speed lines 6 and 7 which can cause instability in iterations. If a map is chosen for a component it is subsequently scaled up or down to match with design point. For example, for a compressor map the operating point is given by three coordinates – the non-dimensional mass flow, the pressure ratio and the corrected rotational speed. However there is one more coordinate in engine model

input file that specifies the position of operating point and the surge line. It is the reverse surge margin and it is defined as follows:

$$Z = \frac{PR_{DP} - PR_{\min}}{PR_{surge} - PR_{\min}} \qquad \text{E2.4.13}$$

With defined surge margin and corrected rotational speed, pressure ratio, corrected mass flow and isentropic efficiency are read from the map and scaled to match values from input files. Similar approach is used for turbine where scaling factors are applied to corrected mass flow and turbine isentropic efficiency.

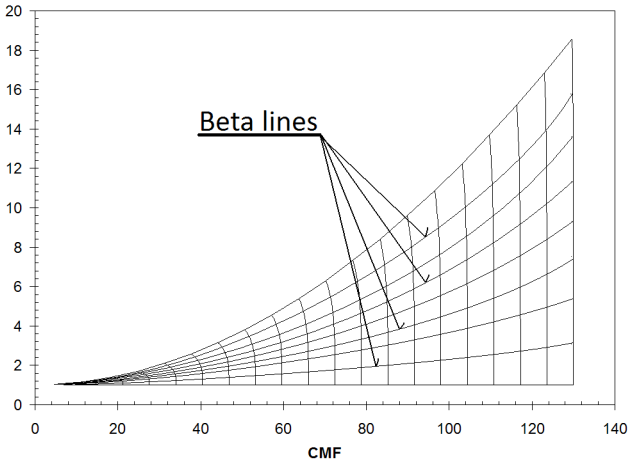


Figure 2.1: Every constant-speed curve is defined by a number of points that is the same for every constant-speed curve in a map. These points connected horizontally constitute β-lines which suit well for map interpolation and iteration purpose (Kurzke 1996, Kong, Ki & Kang 2004). Beta lines application in compressor operating point guess can be also found in appendix 2.9

2.4.3 Steady State Engine Performance Modeling Methodology

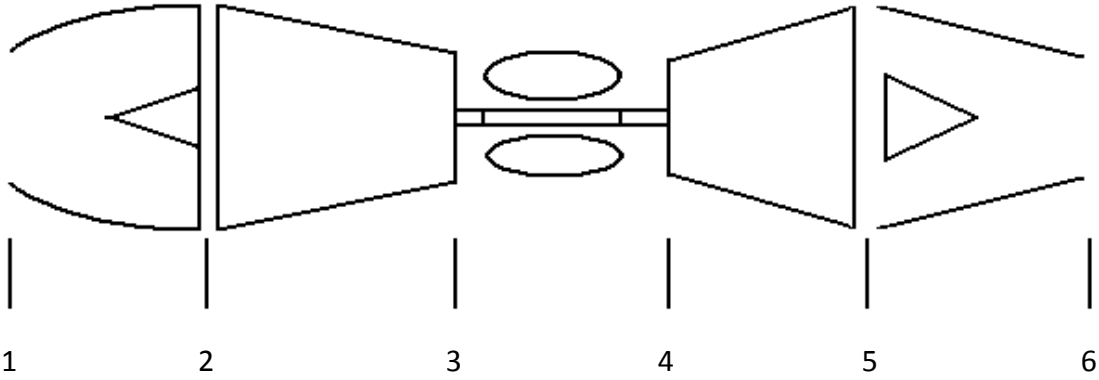
This chapter briefly explains how steady state engine model is processed in Simgaen and Turbomatch. The term *steady state* refers to an operational point where all engine parameters are balanced.

Design point

The first steady state calculation which is run in both programs fixes the engine design. It is referred to as the *design point*. After first design user can carry out either a set of further design point calculations for parametric analysis and for engine optimization,

or a set of *off-design* calculations where engine design is fixed and operating conditions are varied. Engine geometry such as compressor VGV, turbine NGV and variable nozzle area are also allowed to change in off-design. Engine components (intake, compressor, combustor, etc.) are in programs referred to as *bricks*. Every brick has one inlet, one outlet, and the essence of brick is to calculate parameters at the outlet if inlet parameters are known, or guessed by ErrProc subroutine. Some bricks that allow fluid flow separation or flow mixing are allowed to have more than one inlet or outlet. Bricks are connected to each other so that the outlet of an upstream brick becomes inlet for following brick. This connection is called *a station* and it is labeled with a digit. Turbomatch stations are labeled arbitrarily in the input file using an integer and labeling does not need to follow normalized designation as suggested in SAE AS755 (1997) document. For the sake of program simplicity the sequence of bricks in Simgaen is fixed and it allows turbojet (Figure 2.2) simulations only.

The main program calls a subroutine Enginecalc that calls individual subroutines for each component. In Turbomatch the engine configuration is arbitrary; the sequence of individual components is given by user in Turbomatch input file. Program calls corresponding subroutines from the main subroutine TURBOMAT. As mentioned earlier design point analysis is a direct calculation, which means both programs call component subroutines only once according to component sequence. The only exception is an engine model containing heat-exchanger. Heat-exchangers are modeled by means of two bricks of which one represents the cold flow and one hot flow. For a recuperated cycle the cold flow brick is generally placed after compressor and hot flow brick after turbine. To calculate cold flow parameters, hot flow parameters are needed, but they are not known at that point of time. Therefore hot flow parameters are guessed and the solution is obtained by iteration. This process is described in separate chapter in more details.



Engine component sequence:
1-2 Intake
2-3 Compressor
3-4 Combustion chamber
4-5 Turbine
5-6 Propelling nozzle

Figure 2.2: Simgaen turbojet schematic plan with station numbers

Off-design simulation

Off-design simulation uses *thermodynamic matching model* where the solution can be obtained by iteration. The engine design has been fixed now and the user is only allowed to vary the operating conditions, blade angles and a handle (fuel mass flow, COT or HPC corrected rotational speed). The essence of thermodynamic matching is to find values of non-dimensional parameters of one component matched to parameters of other components. Then selected group of non-dimensional parameters represent a point on a component map that is unique for every operating condition. Thermodynamic matching cannot be resolved by direct calculation as it is possible for design point analysis because many variables are unknown, such as compressor pressure ratio, corrected inlet mass flow, rotational speed, etc. The chapter 2.4.4 describes the process of solving one spool turbojet model in Simgaen in more details. The process is equivalent for Turbomatch. List of off design variables that are needed to be guessed and corresponding thermodynamic matching equations of Turbomatch are in appendix 2.10.

2.4.4 Off-design Procedure for One-spool Turbojet

The off-design calculation is an indirect procedure since there are several variables that cannot be calculated directly and their value is guessed. For a turbojet it is the inlet mass flow and the compressor pressure ratio. They are initiated with values from previous solution. Then the parameters of all engine components are calculated and a check is made if compatibility equations are fulfilled. If not, an error is produced, processed by ErrProc function to provide new guesses for variables. Solution is found if the error lies in the externally-defined error band:

Intake

Inlet parameters for intake are either known from ambient parameters that have been acquired from ISA conditions according to the altitude of the operation. Utilized ISA data were obtained from National Oceanic and Atmospheric Administration (1976), which defines the table of pressure, temperature, density and other parameters of the atmosphere. The document corresponds to ISA document (International organization for standardization) up to 32 000 m altitude.

Known parameters:

- M_0 – Inlet Mach number
- T_0 – Ambient static temperature
- p_0 – Ambient static pressure

$$r = 8314.3 \ \frac{J}{mol \times K}$$

1. Intake air velocity:

$$R_a = \frac{r}{m_{wa}}$$

m_{wa} – Molecular mass of air

$$c_{Pa} = f(T_0)$$

$$\gamma_a = \frac{c_{Pa}}{c_{Pa} - R_a}$$

$$V_0 = M_0 \times \sqrt{\gamma_a \; R_a \; T_0}$$

2. Inlet total temperature:

$h_{t0}(T) = h_0(T) + \frac{V_0^2}{2} \Rightarrow$ T_{t0} is acquired by an iterative process, where the initial guess is: $T_{t0} = T_0 + \frac{V_0^2}{2\, c_{Pa}}$

3. Outlet total temperature:

$T_{t2} = T_{t0}$ (no work addition or extraction takes place in intake)

4. Inlet total pressure:

p_{t0} is obtained by an iteration of: $\int_{T_0}^{T_{t0}} \frac{c_{Pa}}{T_t} dT_t = R_1 \times \ln \frac{p_{t0}}{p_0} = R_1 \times \ln \pi_c = \Phi_{t0} - \Phi_0$

with an initial guess: $p_{t0} = p_0 \left(1 + \frac{\gamma_a - 1}{2} \times M_a^2 \right)^{\frac{\gamma_a}{\gamma_a - 1}}$

5. Outlet total pressure:

If the η_{is_i} (isentropic efficiency) of the intake is known, then $T_{t2isent}$ can be obtained and p_{t2} can be calculated by an iteration of:

$$\int_{T_{t0}}^{T_{t2\_isent}} \frac{c_{Pa}}{T_t} dT_t = R_1 \times \ln \frac{p_{t2}}{p_{t0}} = R_a \times \ln \pi_C = \Phi_{t2\_isent} - \Phi_{t0}$$

6. Outlet mass flow:

$$W_2 = W_0$$

Compressor

Known parameters:

- Thermodynamic parameters at inlet known from upstream brick.

To calculate parameters at outlet five non-dimensional parameters must be known:

$$\frac{p_{t3}}{p_{t2}}, \frac{T_{t3}}{T_{t2}}, \frac{W_2 \times \sqrt{R_2 T_{t2}}}{\sqrt{\gamma_2}\, p_{t2}}, \eta_{is\_c}, \frac{N}{\sqrt{\gamma_2 \; R_2 \; T_{t2}}}$$

There are two component maps and one correlation from which three of the parameters can be calculated:

- One PR vs. CMF compressor maps $\rightarrow \frac{N}{\sqrt{\gamma_2 R_2 T_{t2}}}$
- One efficiency vs. CMF compressor maps $\rightarrow \eta_{is\_c}$
- $\int_{T_{t2}}^{T_{t3}} \frac{c_{Pa}}{T_t} dT_t = R_2 \times \ln \frac{p_{t3}}{p_{t2}} = \Phi(T_{t3}) - \Phi(T_{t2})$ with E2.4.9 $\rightarrow \frac{T_{t3}}{T_{t2}}$

Unsolved parameters remain as variables in the iteration process of thermodynamic component matching. Obtaining outlet total temperature and pressure if temperature and pressure ratios are known (or guessed) is straightforward. There is no air bleeding considered, hence $W_3 = W_2$

Combustion chamber

Known parameters:

- Thermodynamic parameters at inlet known from upstream brick.
- COT - Off-design handle (control parameter)
- $\frac{\Delta p_B}{p_{t3}}$ - Combustion chamber pressure losses
- η_B - Combustion efficiency

1. Outlet total pressure:

$$p_{t_4} = p_{t3} \cdot \left(1 + \frac{\Delta p_B}{p_{t3}}\right)$$

2. Outlet total temperature:

$$T_{t4} = COT$$

3. Outlet mass flow and fuel-to-air ratio:

FAR is calculated from energy equation:

$$h_{t3} + FAR \times (LHV\ \eta_b + h_{tff}) = (1 + FAR) h_{t4} \qquad \text{E2.4.14}$$

Outlet mass flow is then:

$$W_4 = (1 + FAR) \cdot W_3$$

Turbine

Known parameters:

- Thermodynamic parameters at inlet known from upstream brick.

Similarly to compressor non-dimensional parameters are used for calculation of outlet parameters:

$$\frac{p_{t5}}{p_{t4}}, \frac{T_{t5}}{T_{t4}}, \frac{W_4 \times \sqrt{R_4 T_{t4}}}{\sqrt{\gamma_4}\, p_{t4}}, \eta_{is\_T}, \frac{N}{\sqrt{\gamma_4\ R_4\ T_{t4}}}$$

Non-dimensional speed and mass flow are evaluated from known inlet parameters. The difference of turbine inlet and outlet enthalpies, dH, is calculated from power balance between compressor and turbine:

Compressor work = Turbine work

$$W_2\left(h_{t3} - h_{t2}\right) = \eta_m \times W_4\left(h_{t4} - h_{t5}\right)$$

$$dH = W_4(h_{t4} - h_{t5})$$

Two component maps and one equation enable to calculate the rest of unknowns:

- dh vs. CMF component maps $\rightarrow \frac{T_{t5}}{T_{t4}}$
- isentropic efficiency vs. CMF component maps $\rightarrow \eta_{is\_T}$
- $\int_{T_{t4}}^{T_{t5}} \frac{c_{Pa}}{T_t} dT_t = R_4 \times \ln \frac{p_{t5}}{p_{t4}} = \Phi_{t5} - \Phi_{t4}$ with E2.4.9 $\rightarrow \frac{p_{t5}}{p_{t4}}$

No turbine cooling is assumed, therefore the outlet mass flow is the same as inlet mass flow: $W_5 = W_4$

Nozzle

Known parameters:

- Thermodynamic parameters at inlet known from upstream brick.
- η_j

1. Critical nozzle outlet pressure and temperature calculation:

$$\frac{p_{t5}}{p_{cr}} = e^{\left(\frac{1}{R_g}\times\int_{T_{cr}}^{T_{t5}}\frac{c_P}{T_t}dT_t\right)}$$, where $\int_{T_{cr}}^{T_{t5}}\frac{c_P}{T_t}dT_t = \Phi_{t5} - \Phi_{cr}$

Equation 2.4.9 is used to calculate pressure ratio for non-isentropic case. The nozzle is considered to be convergent. Mach number is either lower than unity if actual nozzle pressure ratio is below critical pressure ratio or the Mach number equals one if the nozzle pressure ratio is greater than the critical pressure ratio. A check is made to determine which option occurs.

If the nozzle is unchoked $\left(\frac{p_{t5}}{p_0} \le \frac{p_{t5}}{p_{cr}}\right)$ then:

$p_6 = p_0$

$$\int_{T_6}^{T_{t5}}\frac{c_{Pa}}{T}dT = R_a \times \ln\frac{p_{t5}}{p_6} = \Phi_{t5} - \Phi_6 \rightarrow T_6$$

If nozzle is choked $\left(\frac{p_{t5}}{p_0} \ge \frac{p_{t5}}{p_{cr}}\right)$ then:

$$p_6 = \frac{p_{t5}}{\frac{p_{t5}}{p_{cr}}}, \quad T_6 = T_{cr}$$

2. Outlet mass flow calculation:

$W_6 = W_5$, where

$$\frac{W_5\sqrt{R_5\ T_{t5}}}{\sqrt{\gamma_5}\,p_{t5}} = M_6 \times A_6\sqrt{\frac{T_{t5}}{T_6}\frac{\gamma_6}{\gamma_{t5}}}\frac{p_6}{p_{t5}}$$

Overall engine performance

Jet velocity: $V_6 = \sqrt{2\left(h(T_{t6}) - h(T_6)\right)}$ E2.4.15

Jet Mach number: $M_6 = \frac{V_6}{\sqrt{\gamma_a\ R_a\ T_0}}$ E2.4.16

Nozzle exit area: $$A_6 = \frac{W_0(1+FAR)R_g T_6}{p_6 V_6}$$ E2.4.17

Thrust:

$$THRUST = [W_0(1+FAR)V_6 - W_0 V_0] + A_6(p_6 - p_0)$$ E2.4.18

$$SPECIFIC\ THRUST = \frac{[W_0(1+FAR)V_6 - W_0 V_0] + A_6(p_6 - p_0)}{W_0}$$ E2.4.19

$$TSFC = \frac{3600 \cdot FAR \cdot W_0}{THRUST}$$ E2.4.20

Component matching

From the compressor remained two unknowns:

$$\frac{p_{t3}}{p_{t2}}\ and\ \frac{W_2 \times \sqrt{R_2 T_{t2}}}{\sqrt{\gamma_2}\, p_{t2}}$$ E2.4.21

Then there are two mass flow compatibility equations:

3. Compatibility equation between compressor and turbine:

$$\frac{W_4 \times \sqrt{R_4 T_{t4}}}{\sqrt{\gamma_4}\, p_{t4}} = \frac{W_2 \times \sqrt{R_2 T_{t2}}}{\sqrt{\gamma_2}\, p_{t2}} \times \frac{W_4}{W_2}\frac{p_{t2}}{p_{t3}}\frac{p_{t3}}{p_{t4}}\sqrt{\frac{T_{t4}}{T_{t2}}\frac{R_4}{R_2}\frac{\gamma_2}{\gamma_4}}$$ E2.4.22

4. Compatibility equation between turbine and nozzle:

$$\frac{W_5 \times \sqrt{R_5 T_{t5}}}{\sqrt{\gamma_5}\, p_{t5}} = \frac{W_4 \times \sqrt{R_4 T_{t4}}}{\sqrt{\gamma_4}\, p_{t4}} \times \frac{W_5}{W_4}\frac{p_{t4}}{p_{t5}}\sqrt{\frac{T_{t5}}{T_{t4}}\frac{R_5}{R_4}\frac{\gamma_2}{\gamma_4}}$$ E2.4.23

NOTE (Customization of the procedure, when other components are added)

The described procedure is suitable for simulating an off-design of a single shaft turbojet. But to make it applicable for a full range of GT engines every new shaft will involve two new unknowns that must be solved iteratively and two new equations:

1. The mass flow compatibility equation between the original compressor and added compressor
2. The mass flow compatibility equation between the original turbine and added turbine

For heat exchangers there is no heat equilibrium equation (heat extracted from the substance must equal to the heat, added to another substance plus heat losses).

2.4.5 Simgaen and Turbojet Off-design Engine Model Comparison

A model of simple turbojet has been created in Turbomatch for engine cycle analysis. The created engine configuration corresponded to Simgaen turbojet configuration shown on figure 2.2. Simulated engine has following specification:

Overall pressure ratio	8.8	[-]
Compressor isentropic efficiency	0.85	[-]
Turbine isentropic efficiency	0.86	[-]
Mass flow	100.0	[$kg.s^{-1}$]
COT	1400.0	[K]

Table 2.1: Turbojet model specification

Full engine specifications including selected operating point and chosen component maps are shown in appendix 2.11. The engine operating design point on the compressor and turbine map was optimized so that it is as close as possible to optimal map design point and simultaneously to enable desired simulation in selected operating range (table 2.2).

Quantity	**Min.**	**Max.**
Mach number	0.0	0.8
COT	1000 K (950.0 K)	1750 K

Table 2.2: Simulation operating range of turbojet. Combustor outlet temperature of 950 K was not simulated in Simgaen. Altitude and deviation from ISA temperature are kept constant.

The implementation and testing of Turbomatch 20x20 maps

Default Turbomatch compressor maps are stored in 10x5 size format: 10 speed lines, each defined by 5 points. They graphical representation can be found in appendix 2.7. In Simgaen 20x20 compressor map format is set as default. To allow turbojet simulation comparison the engine component maps should be the same. This has been

an opportunity to test Turbomatch feature that enables using user-selected arbitrary-sized component maps (In Turbomatch Legacy version non-default component maps must be of the same size and format as Turbomatch default maps. They cannot be delivered in separate map file, instead they need to be a part of engine model input file written after simulation selectors. Default Simgaen map (appendix 2.6) has been adapted into Turbomatch map format keeping its 20x20 size unchanged.

Default Turbomatch turbine map number 5 (appendix 2.8) was selected for the simulation in both Turbomatch and Simgaen. Its format must have been adapted to follow Simgaen turbine map format this time.

Results

The figure 2.3 shows the turbojet compressor running line on compressor map. At first sight the running line acquired from Simgaen simulation resembles the running line from Turbomatch simulation shifted closer to compressor surge line and with flatter slope. Especially the off-set to the left is evident, implying that maps either compressor or turbine have used different scaling factors for corrected mass-flow axis which is ultimately leaving behind two distinct engine models – one for Simgaen and the other for Turbomatch.

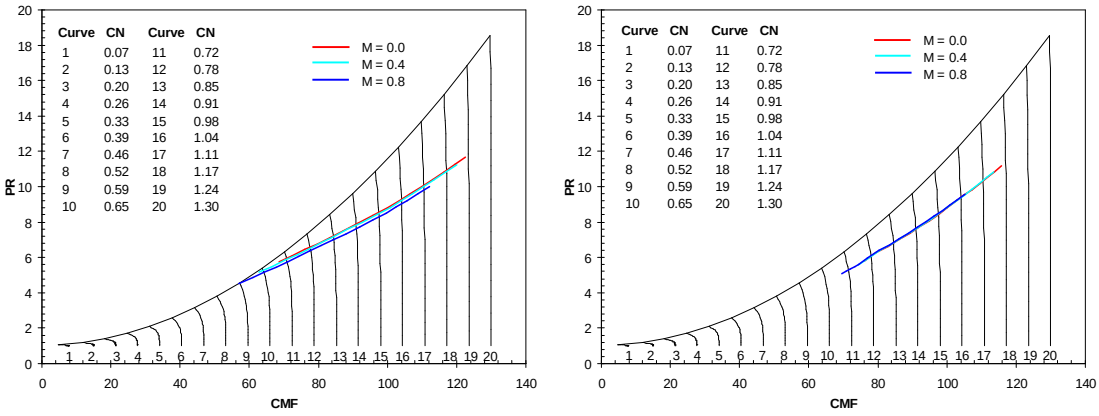


Figure 2.3: Compressor running line – Simgaen simulation (left), Turbojet simulation (right)

Second effect that suggests that turbine mass flow scaling factors are different is observable in appendix 2.12 charts (more precisely A2.12.4, A2.12.5 and A2.12.6)

where the comparison of three thermodynamic parameters (the mass flow, the total temperature and pressure) between the simulations from Simgaen and Turbomatch has been made. Design point parameters that are represented by the green curve match almost exactly. For cases where higher (eventually lover) COT is simulated predominantly the mass-flow differences are considerable, whereas the differences between the total temperature and pressure curves are of much lesser extent. Inconsistent turbine mass flow scaling factors for "the same" engine model are caused by different foundation for turbine mass flow scaling calculation used in Simgaen and Turbomatch (appendix 2.13). Consequently the differences between turbojet performance parameters are higher if they directly depend on engine mass flow (e.g. engine net thrust, appendix A2.12.1). Smaller differences are for specific parameters (e.g. engine specific thrust, appendix A2.12.2).

2.4.6 Conclusion on Simgaen Development.

Turbomatch, having an extensive and perplex structure, is very onerous to work with when it comes to testing and implementing new algorithms, especially if the algorithms include system of non-linear equations and differential equations. Therefore an analogical engine cycle modeling program named Simgaen has been developed which structure is much simpler compared to Turbomatch, but which follows the key algorithms of Turbomatch, so the new algorithms can be implemented and tested on it first. Tests analyzed in chapter 2.4.5 have shown that the program is able to simulate basic Turbojet configuration. Test results revealed minor distinction between Turbomatch and Simgaen scaling procedure which effect is that for the same engine input data a different engine model is actually created. Yet this peculiarity does not cause any obstruction for Simgaen to be used as a testing environment for Turbomatch algorithms.

2.5 The Comparison between the Turbomatch Legacy, Intermediate and TR versions

Chapters 2.2 and 2.3 described the significant changes that Turbomatch underwent throughout its 35 years of existence. The original version of Turbomatch, named here

as the Legacy version, was first modified into the Intermediate version where many structural changes were made, and in 2007 the source code was cleaned up from obsolete code structures and the transient modeling capability was introduced. During these major modifications a few changes were also made (humidity effect calculation, NLE solver adaptation, and more...) which meant that results of the engine model with the same engine design specifications (input data) will not return identical results for Legacy, Intermediate and Transient versions. In this chapter a comparison of the results of the three Turbomatch versions is made. A model of the Rolls Royce Avon 300 Engine was created for the purpose. Its component layout and specifications are found in the engine model input file (appendix 2.14).

The off-design solution of the engine model converged in the Turbomatch Legacy and Transient versions. The convergence failed for the Intermediate version after several trials in which the step in COT was decreased to only 9 K. Therefore only the comparison between the results of the Legacy and Transient version will be shown here.

2.5.1 Solution Analysis

Two types of parameters have been chosen for the comparison between the Turbomatch versions. The choice was influenced by the purpose of the Turbomatch development, which is related to the engine Flight Path Analysis that simulates the engine's operation across a whole flight and provides the inputs necessary to carry out the engine life estimation and shop visit prediction. From a performance point of view the FPA uses net thrust and SFC to calculate flight segment duration and monitors how thermodynamic parameters (temperature, pressure and mass flow) will fluctuate during a flight. Apart from these parameters, a compressor running line was plotted to compare the off design non-dimensional behavior of the engine (figure 2.4)

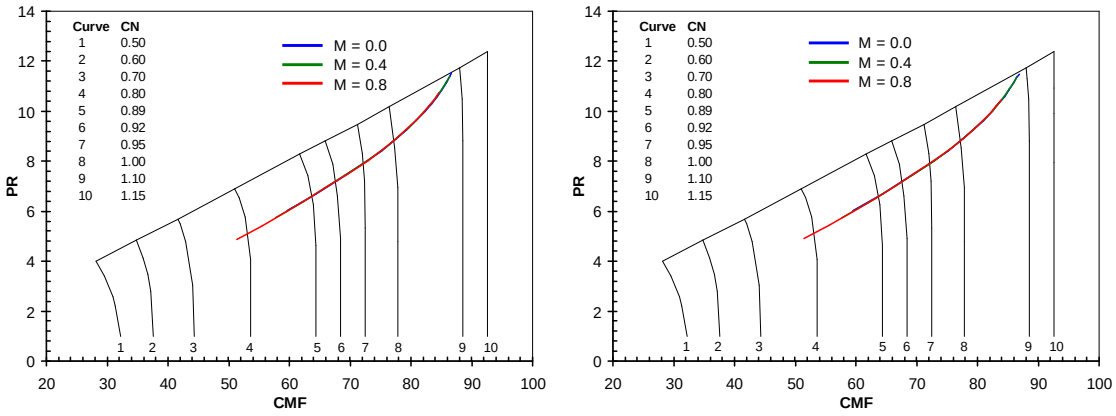


Figure 2.4: Compressor running line comparison of RR Avon 300 engine model (Sea level, dT_{ISA} = 0 °C). The left-had chart acquired from the Turbomatch Legacy version; the right-hand chart comes from the Transient version.

The comparison of compressor running line is essential since once the design of the engine is fixed, all possible operating points of the engine should lie on the line, unless the geometry of the engine changes (e.g. when compressor VGV are employed), or when the engine thermodynamic cycle is affected by external incidents, such as shaft work extraction or air extraction. Figure 2.4 shows two almost identical compressor running lines. The differences between them are not perceptible, because of their small order of magnitude. The differences are unavoidable because of the changes that Turbomatch underwent since 1995. Figure 2.5 shows the turbojet thrust variation. The comparison of other engine parameters is presented in appendix 2.15.

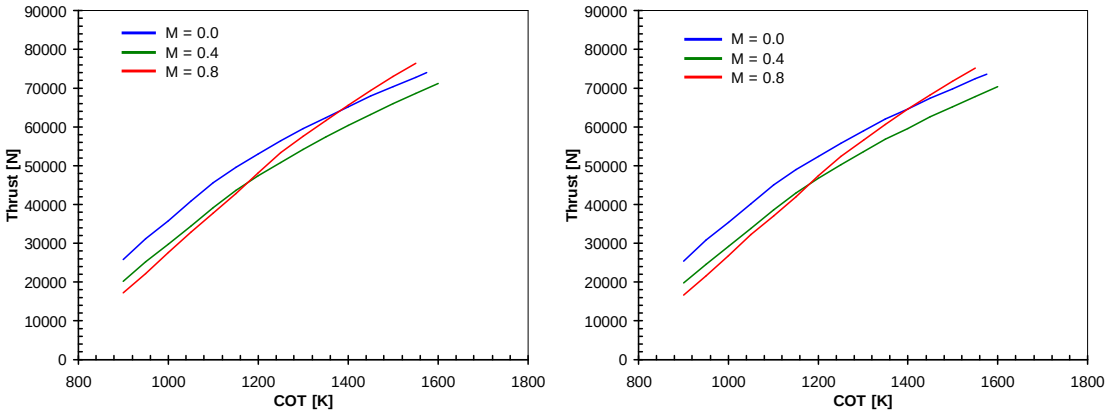


Figure 2.5: The thrust variation of RR Avon 300 engine off design model (Sea level, dT_{ISA} = 0 °C). Left: Turbomatch Legacy version; Right: Turbomatch Transient version.

2.5.2 New Methods in the Turbomatch TR Version

The comparison of the Turbomatch TR to an older one carried out on the Avon 300 engine model off design solution has shown that its results differ from Legacy version results, sometimes up to a few tenths of a percent. These differences were due to several new features and methods that were either added into Turbomatch, or else replaced the old one.

Module WorkingFluidProperties

A module is one example of a feature that was introduced into the Turbomatch Intermediate version. It comprises functions for gas properties calculations, such as enthalpy, thermal phi function, entropy, specific heat capacities, etc. In the Legacy version these functions are scattered in the source code of BRICKSB.FOR file. In the Intermediate version these functions were modified to include the humidity and the combustor water injection effects into gas properties. The gas properties functions of these two versions are no longer equivalent and they will return different values for a fluid with the same intensive thermodynamic properties. The module was transferred to the Turbomatch Transient version without any modifications that would involve changes in calculation procedures.

An example of thermodynamic calculation of compressor outlet total temperature during design point demonstrates how different gas properties functions lead to different outlet temperature result: E2.4.8 and E2.4.9 theoretically explain how outlet total temperature of compressor can be evaluated if the pressure increases and the isentropic efficiency is known:

The phi function is calculated from a pressure ratio, inlet temperature and fuel-to-air ratio (which equals to zero for this case) and its value is stored in the variable named PHIBS. The outlet temperature referring to an isentropic process is calculated from E2.4.8 and is stored in the TBI variable. Knowing the inlet and outlet static temperature appertaining to selected pressure ratio for an isentropic process, an isentropic enthalpy difference is evaluated. E2.4.8 is consequently used to evaluate the enthalpy

difference for a non-isentropic process (ENTHB1 - ENTHA) using compressor efficiency from which the ultimate outlet total temperature B1%T can be determined. For better clarification the corresponding source code snippet calculating the compressor outlet total temperature is shown in appendix 2.16. The values of mentioned variables that for RR Avon 300 design point analysis are compared in Table 2.3

Parameter description	Declared in Turbomatch	TM Legacy ver. value	TM Transient ver. value
Compressor inlet total pressure	A%P	1.000000	1.000000
Compressor outlet total pressure	B%P		
Compressor inlet total temperature	A%T	288.1500	288.1500
Thermodynamic phi function of compressor outlet total temperature	PHIBS	25.38550	0.163728
Spec. enthalpy of air at the inlet	ENTHA	68.86951	3.617241
Spec. enthalpy of air at the outlet	ENTHB1	139.6462	75.35893
Compressor outlet total temperature	B%T	578.6828	580.780

Table 2.3: Values of intensive thermodynamic properties and functions to demonstrate differences of results of thermodynamic processes between older and new version of Turbomatch. Parameters were collected from COMPRE brick during RR Avon 300 engine model design point analysis

The table values for enthalpy suggest that the enthalpy functions of Turbomatch Legacy and TR version do not use the same algorithm. Different enthalpy units are not the cause, because if the difference was caused by units the values of ENTHA and ENTHB1 would differ by the same multiplier.

Increased variable precision

A new module PortSet has been introduced. This module specifies the requirements for precision of every real number in Turbomatch to minimum 10 valid digits. The precision is calculated by Fortran intrinsic function selected_real_kind() and is stored in variable RK (Real Kind). Originally all real numbers had only default precision that was 6 valid digits on a 32-bit computer. PortSet module is used by every other module and

routine in the program and every real number is than declared to have the precision of RK kind.

NLE solver update

Legacy Turbomatch version used a modified version of Newton – Raphson method to calculate set of non-linear equations in Turbomatch. An improvement on the algorithm calculating derivatives for the Jacobian matrix has been made. A feature has been implemented that allows guesses to equal zero. The original subroutine for solving the Jacobian matrix has also been replaced by a new one that is claimed to work faster although its algorithm is more complicated as it requires the Jacobian matrix to be split to an upper and lower triangular matrix. Details are found in chapter 2.3.6

2.5.3 Conclusion

An engine model of RR Avon 300 has been created and simulated by two Turbomatch versions – the Legacy and the new Transient version. The results of selected off-design performance and thermodynamic parameters were compared to each other. The results were similar with minor differences. Further investigation of the differences found that several subroutines and functions that have been updated return different values compared to previous version. Some of these routines have been introduced earlier, such as gas properties function and some were introduced during Transient capability implementation. All these updates were implemented to improve the simulation accuracy.

CHAPTER 3

TRANSIENT PERFORMANCE

Throughout its existence, the Turbomatch Legacy version has been used for thermodynamic cycle and performance simulations for gas turbine engine models with arbitrary configurations. It was capable of simulating any kind of operating condition effects. However, all the calculations were only carried out in steady state mode; that is, if the engine parameters are balanced and no changes take place over the time. For real engines such conditions are possible only theoretically as the real engine experiences many operating changes during its operation. Both industrial and aero engines experience changes first when they are started, when the fuel flow increases or decreases, and when they are stopped. Engines, moreover, need to cope with ambient air pressure and temperature changes. Although these changes are more evident for aero-engines as the aircraft climbs or descends, even the industrial engines experience changes in operating conditions due to weather fluctuations and day and night temperature changes. All these changes shift the engine operating point from its original balance to another operating point at which the engine parameters reach a new state of balance. The process by which the engine operating point moves from one balanced condition to another is called the *dynamic response,* during which the engine performance is *transient*. For steady state simulation an assumption is made that ambient parameters and engine control parameters (fuel flow, variable geometry, etc.) are steady for a long period of time.

A new transient performance algorithm was implemented into the source. With transient performance Turbomatch is able to simulate e.g. take-off, reverse thrust, missile launch (instant Mach number and ambient air temperature change) industrial engine load change, which may also involve a combined cycle (Chilvers, Milanović

2002). Thus, it is possible to simulate any phenomena when a sudden (or gradual) change in ambient or control parameters occurs. Transient performance simulation provides a deeper understanding of what is really happening inside the engine during the take-off segment or reverse thrust in flight path analysis and contributes to more accurate engine life estimation (Hariharan 2009). The algorithm is also useful to simulate transient working line deflection to enable the control system to maintain the fuel flow schedule such that the engine will not run into risky operating conditions (e.g. where the compressor working line would cross the surge margin).

Early engine control systems have used simplified transient performance models where limits of parameters were plotted against non-dimensional parameters. For this purpose placing the parameters in groups has proved to be a very sensible step because when referred fuel flow and referred speed are fixed during transient operation, then all referred non-dimensional parameters are also fixed (Walsh and Philip, 2004). The advantage of the method was its speed. The control system simply used a look-up table to determine the fuel flow (eventually guide vanes angle) according to measured data from the engine. This method was only suitable for control systems because its purpose was to control the engine operation, rather than investigating the engine's behavior during dynamic response. For the engine thermodynamic cycle modeling a mathematical approach has to be used. The foundations of transient behavior modeling using thermodynamic matching model are explained by Fawke and Saravanamuttoo (1971, 1973). Their proposed model used a so-called one dimensional approach (incidentally, Turbomatch Legacy version also uses one-dimensional methods). Yin (2000) created a two dimensional transient performance model of a large bypass turbofan engine and made a comparison with the one-dimensional model. Although the two-dimensional method produces results with higher accuracy, it is not suitable for general simulation because it can only simulate the designed engine. The one-dimensional method offers the flexibility of simulating any engine configuration, created simply by connecting bricks of corresponding components together and inputting the engine specifications.

During the research of the Transient performance modeling, two transient methods have been tested:

- Rapid transient performance method
- Thermodynamic matching transient method

Both transient performance methods use component bricks to build the engine model. They require initial values for simulation taken from previous steady state calculation. After steady state the program switches to transient mode and sets the simulation time to:

$$t = t_{i=0}$$

When dynamic response of the engine model for the initial time is calculated the program increments the time with a user-selected time step and the process is repeated until the simulation time reaches the final time. The time increment chosen must be small enough to allow volume dynamics implementation. The longer the time step the faster the engine simulation, but with large time steps the mass in the volume increases too much and convergence of simulation might be not achieved. When the power produced by the turbine equals the power needed to drive compressors on the same shaft in transient analysis, this balance is violated. If the turbine produces more power than is required to drive the compressor, the shaft accelerates until a new balance point between the compressor and turbine power is found. Vice versa, if the compressor uses more power than the turbine is able to deliver, deceleration of the shaft occurs. The difference between the turbine and the compressor, the surplus power, is obtained from:

$$SP = \eta_m \times W_4 \left(h_{t4} - h_{t5}\right) - W_2 \left(h_{t3} - h_{t2}\right) \qquad \text{E3.0.1}$$

The total inertia of the shaft, consisting of turbine and compressors driven by that turbine, acts against any change in shaft rotational speed:

$$SP = M \times \omega = I \times \alpha \times 2\pi N = I \times \frac{d\omega}{dt} \times 2\pi N = \frac{4\pi^2 I\ N\ dN}{dt} \qquad \text{E3.0.2}$$

After a transposition a change of rotational speed is obtained:

$$\frac{dN}{dt} = \frac{SP}{4\pi^2 I\ N} \qquad \text{E3.0.3}$$

In the model a differential equation E3.0.3 is resolved for every shaft numerically. The new value of rotational speed is theoretically given by:

$$N = \int_{t}^{t+dt} \frac{SP}{4\pi^2 I\ N}\, dt \qquad \text{E3.0.4}$$

In the faster method the increment to rotational speed is evaluated by explicit Euler integration (NATO 2007):

$$N_{next} = N_{now} + \frac{dN}{dt} \cdot \Delta t \qquad \text{E3.0.5}$$

Whereas in the transient performance method involving the thermodynamic matching the implicit Euler integration was used, because it leads to a more stable solution:

$$N_{now} = N_{last} + \left(\frac{dN}{dt}\right)_{now} \cdot \Delta t \qquad \text{E3.0.6}$$

Here instead of predicting the rotational speed to the future, the increment to rotational speed is evaluated for actual time step.

NOTE: Both methods have been implemented and tested on Simgaen before they were implemented in Turbomatch. Simgaen implementation is not discussed here. Instead the implementation to Turbomatch is explained in details.

3.1 Volume Dynamics

During steady state simulation it is assumed that for every component the inlet mass flow equals the outlet mass flow. For such case this assumption is perfectly valid since the matter (in this case air or gas) cannot be created or disappear. When the engine operates in transient conditions this rule is no longer valid, especially for components such as combustor or turbine duct, which volumes are greater compared to other components. For example, if the fuel flow to the combustor is suddenly increased the turbine entry temperature is also increased. Assuming that turbine operates under choked conditions its inlet and outlet non-dimensional mass flow is constant:

$$\frac{W_{in} \times \sqrt{R_{in} T_{t\,in}}}{\sqrt{\gamma_{in}}\, p_{t\,in}} = const. \qquad \text{E3.1.1}$$

$$\frac{W_{in} \times \sqrt{R_{in} T_{t\,in}}}{\sqrt{\gamma_{in}}\, p_{t\,in}} = const. \qquad \text{E3.1.2}$$

considering that $R_{in} = R_{out}$ and neglecting the influence of changing γ:

$$\frac{W_{in} \times \sqrt{T_{t\,in}}}{W_{out} \times \sqrt{T_{t\,out}}} \times \frac{p_{t\,out}}{p_{t\,in}} = const. \quad \text{and} \quad \frac{T_{t\,in}}{T_{t\,in}} = \left(\frac{p_{t\ in}}{p_{t\ out}} \right)^{\frac{\gamma-1}{\gamma}} \qquad \text{E3.1.3}$$

ultimately:

$$\frac{T_{t\,in}}{T_{t\,out}} = const. \qquad \text{E3.1.4}$$

The temperature ratio across the turbine is constant for an ideal isentropic case and constant gas properties.

The power generated by the turbine is the function of mass flow and inlet and outlet turbine temperature:

$$TP = W_{in} c_p \left(T_{t\,in} - T_{t\,out} \right) = W_{in} c_p\ T_{t\,in} \left(1 - \frac{T_{t\,out}}{T_{t\,in}} \right) \qquad \text{E3.1.5}$$

Because the ratio of turbine inlet and outlet temperature is constant the increase in turbine power only depends on increase of turbine inlet temperature in a turbine operating in choked region. To keep the non-D mass flow of the turbine constant the pressure in the combustion chamber must increase by a square root of the turbine inlet and outlet temperature ratio increase. Having in mind the compressor power equation:

$$CP = W_{in\ C} \cdot c_p \left(T_{t\,out\,C} - T_{t\,in\,C} \right) \text{ and } \frac{T_{t\,out\,C}}{T_{t\,in\,C}} = \left(\frac{p_{t\,out\,C}}{p_{t\,in\,C}} \right)^{\frac{\gamma-1}{\gamma}} \qquad \text{E3.1.6}$$

Will result, that the compressor power will also rise, but not by the same rate, as the turbine power. And the surplus power, obtained form the difference between the turbine and compressor makes the shaft accelerating and moves the operation point of the compressor on the compressor map to a new position.

This example is very theoretical and it was introduced to illustrate the effect of sudden change in engine control parameter. In engine model that represents the real gas turbine engine the turbine non-dimensional inlet mass flow needs to be evaluated from the turbine map (Figure 3.1). Especially in low-speed operation the turbine is not necessarily choked.

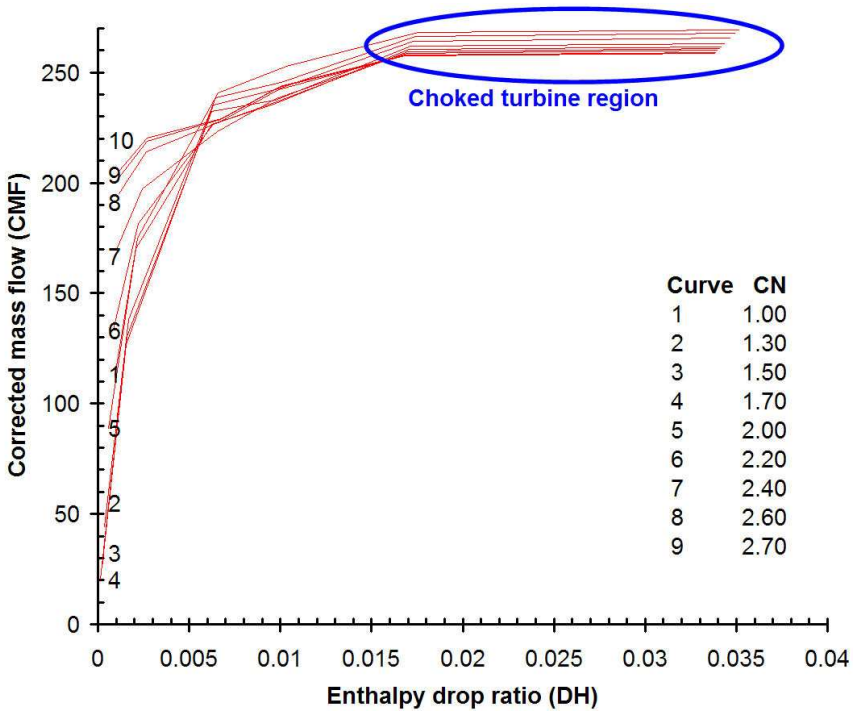


Figure 3.1: A typical turbine map depicting the region of choked operation

As said before, the pressure in the combustion chamber will increase. This increase is caused by an increase in mass flow entering the combustor. Hence, the bigger the combustor volume, the longer the change of pressure in the combustion chamber. This effect is called the volume dynamics.

Transient performance methods can be also divided according to criterion whether they include the volume dynamics, or not:

- The Constant Mass Flow method
- The Intercomponent Volume method

3.1.1 Constant Mass Flow Method

The method always assumes that the mass flow of the air (or gas) entering the component must equal to mass flow leaving the component. The advantage of this approach is that the time interval (time increment of the iteration) can be much longer, than by ICV method and therefore the analysis could be performed faster. This was very important in the past, when the real time simulation of the transient was necessary, but computers were not that fast. This method also suits for simulating engines with smaller volumes of components. In fact the results of this method are very similar to results, obtained from the ICV method, but the main difference occurs in the first part of the transient running line on the compressor map. Generally is the method very similar to the ICV method, therefore it is not described here, but the differences from the ICV method, are outlined in the ICV method simulating process description.

3.1.2 Intercomponent Volume Method

This method is more accurate, than the CMF method, because the volume dynamics is included (Fawke, Saravanamuttoo 1971). The accuracy of the method rises with more accurate estimated volumes of components. If the GT engine incorporates bulky components, the differences between results of CMF and ICV method become more significant and the importance of using ICV method rises.

To capture the physics behind the volume dynamics, the component is separated in two segments, as seen on figure 3.2.

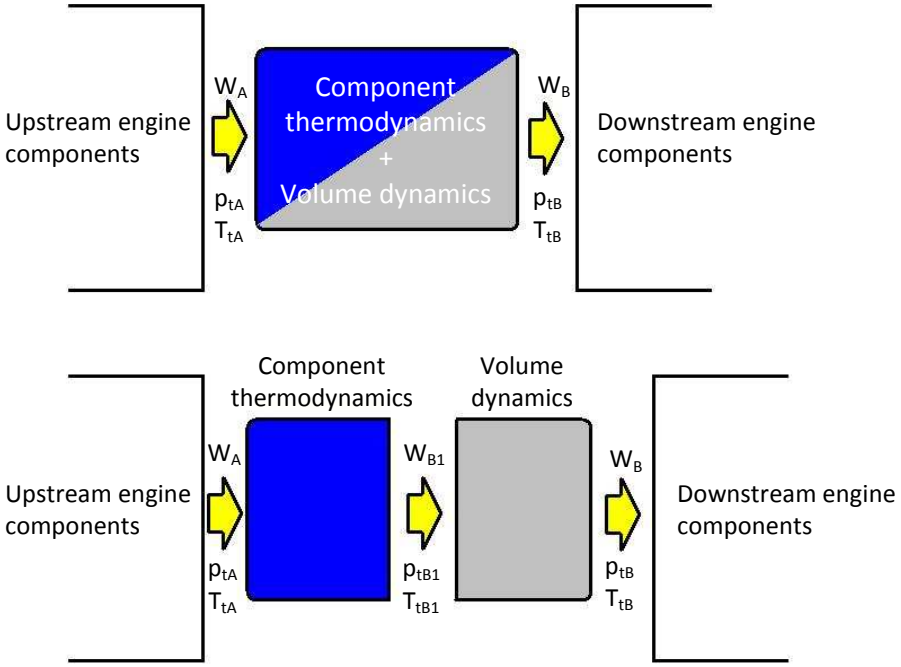


Figure 3.2: Intercomponent method fundamentals – splitting the component into two segments. The subscript B1 describes component intermediate station – that is before any volume dynamics took place.

The first segment is assumed to retain the same properties at steady state calculation. Therefore the inlet mass flow $W_A = W_{B1}$. The second segment calculates the volume dynamics only. In this segment $W_{B1} \neq W_B$ and any surplus mass flow whose value is given by the difference between inlet and outlet mass flow contributes the change in fluid mass in the volume component. The overall mass of the fluid stored in the component during steady state simulation and at initial time of transient simulation is given by:

$$m_{vol} = \frac{P_{vol}}{R_{fluid} \cdot T_{vol}} \cdot Vol_{component} \qquad \text{E3.1.7}$$

Because the balance between inlet and outlet mass flow has been broken, the mass of the fluid inside the component starts to change.

The new mass of the fluid after the time is incremented by time step Δt is calculated from the mass flow difference:

$$\frac{dm_{vol}}{dt} = \left(W_{in} - W_{out}\right)$$

$$m_{vol\ new} = \left(W_{in} - W_{out}\right)\Delta t + m_{vol\ old} \qquad \text{E3.1.8}$$

Now the new pressure inside the volume component is evaluated from:

$$p_{new} = \frac{R_{new} \cdot T_{new}}{Vol} \cdot m_{vol\ new} \qquad \text{E3.1.9}$$

Thus the new pressure of the component is known

3.1.3 Heat Storage

Except for storing the surplus fluid the component volume has also ability to store surplus heat. During transient simulation, the enthalpy entering the component does not necessarily equal to enthalpy leaving the component. The difference between the inlet and outlet enthalpies, the surplus heat, will affect the temperature of the fluid in the component (Pilidis 2006). Heat storage calculation begins with time t_0:

Simulation time: t = t_0

The heat storage is analyzed on thermodynamic system consisting of three fluid elements (Figure 3.3). First element represents the fluid with enthalpy flowing into the component volume:

$$m_{in} \cdot h\left(T_{in}\right)$$

The enthalpy stored in the component volume is expressed by:

$$m_{vol\ old} \cdot h\left(T_{vol\ old}\right)$$

It contains the fluid element with enthalpy, that remains in the component after the simulation time is incremented by Δt:

$$m_{vol\ old} \cdot h\left(T_{vol\ old}\right) - m_{out} \cdot h\left(T_{vol\ old}\right)$$

Third element represents the fluid with enthalpy, that leaves the component after the time is incremented:

$$m_{out} \cdot h\left(T_{vol\ old}\right)$$

The temperature inside the component volume is assumed to be homogenous.

The mass of the separate elements is calculated using the mass equation E3.1.9:

$$m_{in} = W_{in} \cdot \Delta t$$
$$m_{out} = W_{out} \cdot \Delta t$$

The mass of the fluid in the volume component is evaluated from E3.1.7.

Simulation time: $t = t_0 + \Delta t$

After the time increment the fluid of the first element has entered the volume component and the fluid of the third element left the component (Figure 3.4). The new enthalpy, stored in the component is:

$$m_{in}\, h(T_{in}) + m_{vol\ old}\, h(T_{vol\ old}) - m_{out}\, h(T_{vol\ old}) = m_{vol\ new}\, h(T_{new\ vol})$$

And the enthalpy of the fluid that left is:

$$m_{vol\ old} \cdot h\left(T_{vol\ old}\right)$$

Because the thermodynamic system is assumed to be isolated, its overall enthalpy is constant:

Enthalpy of system$_{t=0}$ = Enthalpy of system$_{t=0+\Delta t}$

$$m_{in} \cdot h\left(T_{in}\right) + m_{vol} \cdot h\left(T_{vol\ old}\right) - m_{out} \cdot h\left(T_{vol\ old}\right) + m_{out} \cdot h\left(T_{vol\ old}\right)$$
$$=$$
$$m_{vol\ new} \cdot h\left(T_{vol\ new}\right) + m_{out} \cdot h\left(T_{vol\ old}\right)$$

After transposition:

$$h(T_{vol\ new}) = \frac{1}{m_{vol\ new}} \left[m_{in} \cdot h\left(T_{in}\right) + m_{vol} \cdot h\left(T_{vol\ old}\right) - m_{out} \cdot h\left(T_{vol\ old}\right) \right] \qquad \text{E3.1.10}$$

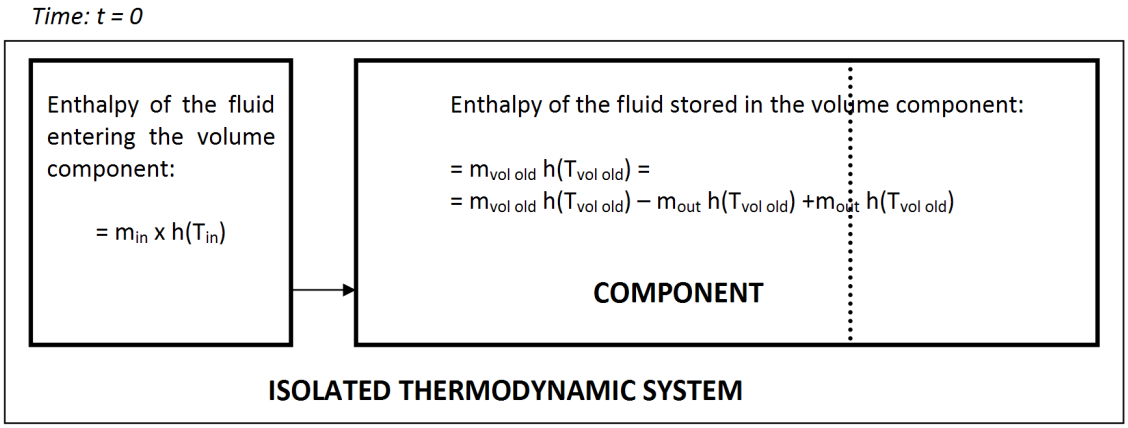


Figure 3.3: A schema of thermodynamic system consisting of three fluid elements, of which one is about to enter the component volume and two are stored in the component volume

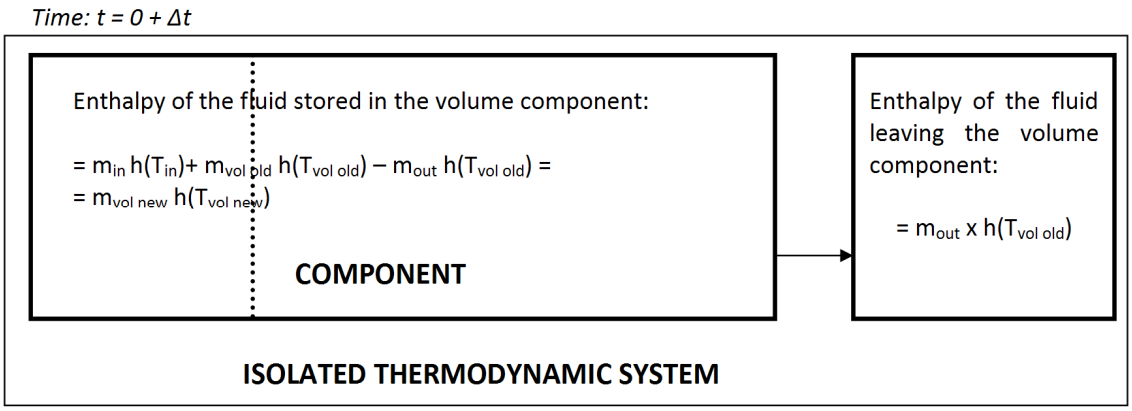


Figure 3.4: A schema of thermodynamic system consisting of three fluid elements after a time increment. First element has entered the volume component and the last has left it.

The new temperature of the fluid in the volume component can be evaluated from component enthalpy E3.1.10 using Turbomatch enthalpy function. The temperature of the fluid leaving the component equals to the temperature of the fluid of the component before any fluid of the first element entered.

3.2 Rapid Transient Performance Method

This method was initially chosen for Turbomatch. The detailed explanation of the method is found in (MacIsaac, Saravanamuttoo 1974). Transient performance simulation starts with a steady state analysis to determine the engine thermodynamic state at the beginning of Transient. This can be the design point, but in most cases it is off-design. Then the program initiates the process of transient performance calculation.

In this method the solution is not obtained by a numerical iteration that solves system of non-linear equations. The iterations slow the simulation significantly, because the solver needs to call the engine loop for number of times that is greater than number of variables, since it requires to calculate the original values of equations and the values for the Jacobian matrix (chapter 2.3.6). For instance the average two-spool turbofan engine model has 10 non-linear equations with ten unknown variables that need to be solved. It means that solver needs to call the entire engine loop for more than 10 times to produce a value for a second guess for the unknown variables. This process may repeat 3-12 times until they satisfy all equations are found, thus calling the engine brick sequence sometimes even for one-hundred times or more just to find one operating point.

The rapid transient performance method is an adaptation of real time aerothermal transient performance model (Walsh, Philip 2004; MacIsaac, Saravanamuttoo 1974) especially made for Turbomatch. The adaptation was carried out for two reasons:

- To keep the steady state engine components calling sequence
- To be able to create a transient performance model of any configuration
- To be able to use any number of volume components

3.2.1 Original Method

The original method starts at the combustor component where thermodynamic parameters at the inlet and outlet are first evaluated. It is followed by calculations in the first turbine component. Only after that the intake component is called followed by other components in the row (Figure 3.5).

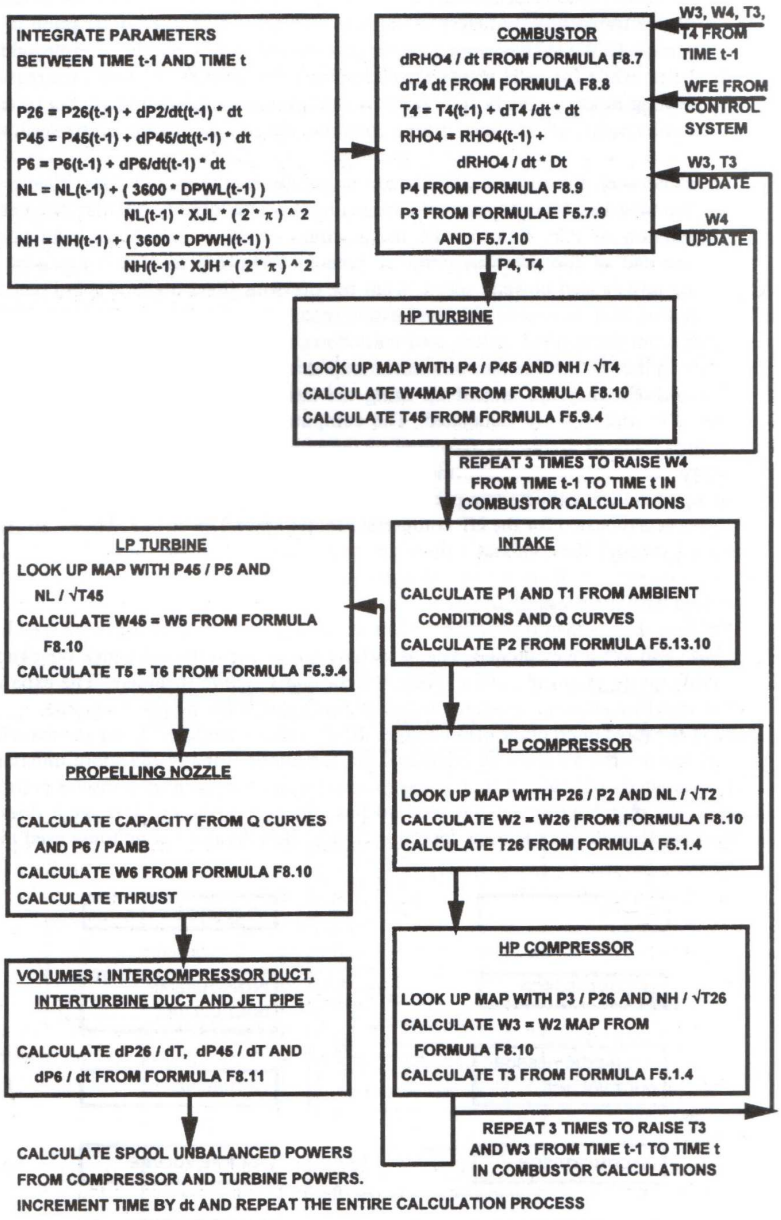


Figure 3.5: Real time aerothermal model proposed in (Walsh, Philip 2004). The calling sequence of the engine model components does not follow the sequence in which the components are ordered.

3.2.2 Turbomatch Adaptation

Rapid transient performance method starts at the intake. The inlet parameters of the intake remain unchanged from the previous step unless they are deliberately selected to change (e.g. the ram pressure at temperature at the inlet changes if the impact of ambient conditions on engine behavior is modeled). The main difference is the way how thermodynamic parameters are evaluated. In steady state analysis all thermodynamic parameters are evaluated downstream the engine model. During transient temperature and pressure are generally evaluated downstream, but the mass flow is evaluated upstream (Figure 3.6).

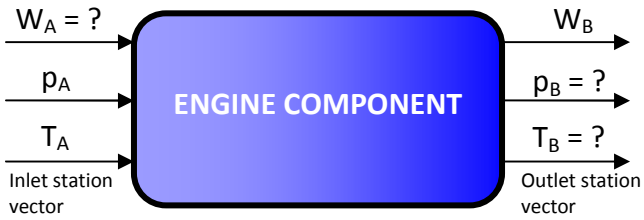


Figure 3.6: Illustration of known and unknown items of station vector of an engine component during transient simulation

In transient simulation all station vector items are calculated down stream of engine model apart from component mass-flow. Inlet mass flow is calculated in the component, whereas the outlet mass flow is known from the succeeding component from a previous time step.

The method is here presented on an engine model of two-spool gas turbine. It demonstrates how basic thermodynamic parameters of the engine (mass flow, pressure and temperature) are evaluated (Figure 3.7).

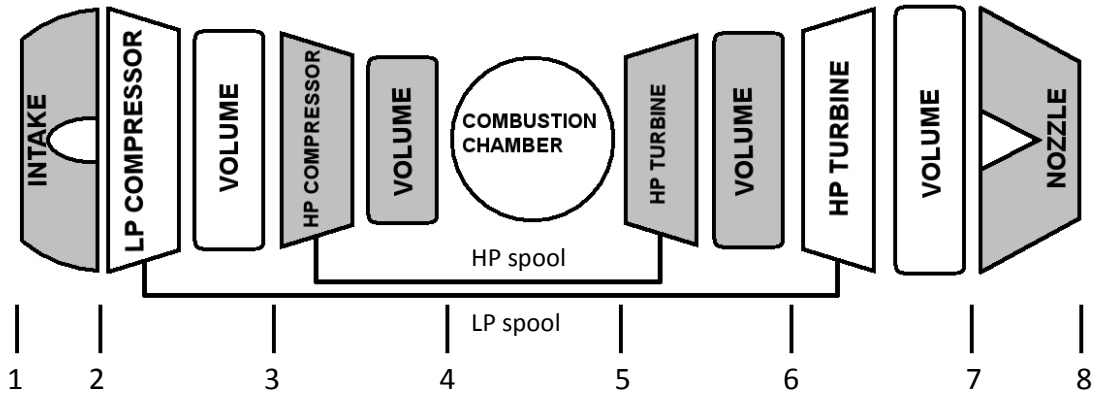


Figure 3.7: Two-spool gas turbine configuration schema. Volumes have been added to include the volume packing effect.

The order in which component bricks are called follows the engine model order. For every brick it is first presented which parameters are known (named as “Inputs”) and which are about to be calculated using brick routines (names as “Outputs). Consequently it is explains how values for outputs are determined.

3.2.3 Rapid Transient Performance Method of a Two-spool Gas Turbine Engine

Transient step initiation

For the first transient step after steady state the simulation time is started from a desired initial time. For all succeeding calculations the simulation time is incremented by a desired time step:

$t = t_{INITIAL}$ (First transient calculation)

$t = t_{previous} + \Delta t$ (All following steps)

Then, all parameters calculated at preceding step as “new” become “old”. This applies particularly to parameters determining the thermodynamic state in the control volumes – the pressure, temperature of the fluid in the volume.

Intake

Inputs:

- W_B Outlet mass flow
- p_{sA} Static pressure at the inlet
- T_{sA} Static temperature at the inlet
- M_A Inlet Mach number
- Δp_{intake} Intake pressure loss

Outputs:

- W_A Inlet mass flow
- p_{tB} Total pressure at the outlet
- T_{tB} Total temperature at the outlet

Total temperature at the outlet:

Evaluated from the enthalpy calculated at the outlet station using energy equilibrium:

$$h(T_{tB}) = h(T_{sA}) + \frac{[V_A(M_A)]^2}{2}$$

Total pressure at the outlet:

The value of total outlet pressure is evaluated from the isentropic equilibrium:

$$\int_{T_{sA}}^{T_{tB}} \frac{c_p}{T} dT = R \ln \frac{(p_{tB} - \Delta p_{\text{int}ake})}{p_{sA}}$$

Where the temperature integral is calculated from the difference of phi-functions:

$$\int_{T_{sA}}^{T_{tB}} \frac{c_p}{T} dT = \phi_2(T_{tB}) - \phi_1(T_{sA})$$

The final value of outlet pressure is obtained after subtracting the pressure loss at the intake:

$$p_{tB} = (p_{tB} - \Delta p_{\text{int}ake})$$

Mass flow at the inlet:

$W_A = W_B$

LP and HP compressor

The same process is applied for every compressor of the engine model.

Inputs:

- W_B Outlet mass flow
- p_{tA} Total pressure at the inlet
- T_{tA} Total temperature at the inlet
- $p_{tB1\ old}$ Total pressure in the volume component from the previous time step
- $T_{t\ vol\ old}$ Total temperature of the fluid in the volume from the previous time step
- PCN Relative rotational speed

Outputs:

- W_A Inlet mass flow
- p_{tB} Total pressure at the outlet
- T_{tB} Total temperature at the outlet
- $p_{tB1\ new}$ Total pressure in the volume component for actual time step
- $T_{t\ vol\ new}$ Total temperature of the fluid in the volume for actual time step
- CP Compressor power

Mass flow at the inlet:

Rotational speed PCN is taken from the previous step. Compressor pressure ratio is defined as:

$$PR = \frac{p_{tB1\,old}}{p_{tA}}$$

The inlet corrected mass flow and the isentropic efficiency values are obtained from the compressor map. Compressor inlet mass flow and volume inlet mass flow are evaluated from the corrected mass flow:

$$W_A = CMF \cdot \frac{p_{tA} \cdot \sqrt{T_{ISA\,SLS}}}{\sqrt{T_{tA}} \cdot p_{ISA\,SLS}}$$

$$W_{B1} = W_A$$

Compressor power:

$$CP = W_A \cdot \left[h\left(T_{tB1}\right) - h\left(T_{tA}\right)\right]$$

Total temperature at intermediate stage:

The difference of phi-functions, from which the isentropic temperature is calculated, is obtained by E2.4.8 where pressure ratio is known. For the final temperature the isentropic efficiency effect is taken into account:

$$h(T_{tB1}) - h(T_{tA}) = \frac{h(T_{tB1})_{is} - h(T_{tA})}{\eta_{C\,is}} \Rightarrow T_{tB1}$$

Total pressure and temperature at the outlet:

After the compressor brick the volume component calculations take place (chapter 3.1.2 and 3.1.3). The mass of the volume fluid is known from the previous step. Its change is calculated from the difference of inlet and outlet mass flow of the component (W_{B1} and W_B). New mass will affect the pressure in the volume which is calculated by E3.1.9. $\Rightarrow p_{B1\,new}$, $p_B = p_{B1}$

The temperature of the fluid leaving the volume is set to the temperature of the fluid in the volume from the previous time step:

$$T_{tB} = T_{t\ vol\ old}$$

The new volume fluid temperature that is going to be used in the next step is acquired from E3.1.10.

Combustion chamber

Inputs:

- W_B Outlet mass flow
- p_{tA} Total pressure at the inlet
- T_{tA} Total temperature at the inlet
- W_{ff} Fuel flow
- η_b Burning efficiency
- $\Delta p/p$ Pressure loss

Outputs:

- W_A Inlet mass flow
- p_{tB} Total pressure at the outlet

- T_{tB} Total temperature at the outlet
- FAR Fuel-to-air ratio

Mass flow at the outlet and fuel-to-air ratio:

$$W_B = W_A + W_{ff}$$

$$FAR = \frac{1}{\frac{W_B}{W_{ff}} - 1}$$

Total temperature at the outlet:

$$h(T_{tB}) = \frac{h(T_{tA}) + FAR \cdot (LHV \cdot \eta_b + h(T_{ff}))}{(1 + FAR)} \quad \Rightarrow \quad T_{tB} \qquad \text{E3.2.1}$$

Eventually if Total outlet temperature is given, the fuel flow can be evaluated from a transposed version of E3.2.1

HP and LP turbine

The same process is applied for every compressor turbine of the engine model.

Inputs:

- W_B Outlet mass flow
- p_{tA} Total pressure at the inlet
- T_{tA} Total temperature at the inlet
- $p_{tB1\ old}$ Total pressure in the volume component from the previous time step
- $T_{t\ vol\ old}$ Total temperature of the fluid in the volume from the previous time step
- $\eta_{T\ is\ old}$ Isentropic efficiency from the previous step
- η_m Mechanical efficiency
- PCN_{old} Relative rotational speed from the previous step
- N_{DP} Shaft rotational speed at the design point
- CP Compressor power (cumulative power of all compressors driven by turbine shaft
- EP Additional power extracted

Outputs:

- W_A Inlet mass flow
- p_{tB} Total pressure at the outlet
- T_{tB} Total temperature at the outlet
- $p_{tB1\ new}$ Total pressure in the volume component for the current time step
- $T_{t\ vol\ new}$ Total temperature of the fluid in the volume for the current time step
- $\eta_{T\ is\ new}$ Isentropic efficiency for the current step
- PCN_{new} Relative rotational speed for the current step
- TP Turbine power

Total temperature at intermediate stage:

Calculated from pressure ratio $p_{tA}/p_{tB1\ old}$ and isentropic efficiency $\eta_{T\ is\ old}$ by E2.4.8 and E2.4.9 $\Rightarrow T_{tB1}$

New turbine isentropic efficiency and mass flow at the turbine inlet and intermediate stage:

Updated values of isentropic efficiency and inlet mass flow are acquired from the turbine map. The PCN_{old} is known from the previous step and the enthalpy drop across the turbine is evaluated from:

$$dh_T = h(T_{tA}) - h(T_{tB1})$$

Corrected mass flow and isentropic efficiency is obtained from the map. For inlet mass flow:

$$W_A = CMF \cdot \frac{p_{tA} \cdot \sqrt{T_{ISA\,SLS}}}{\sqrt{T_{tA}} \cdot p_{ISA\,SLS}}$$

$$W_{B1} = W_A$$

Total pressure and temperature at the outlet:

Thermodynamic parameters at the inlet to the control volume have been determined. Chapters 3.1.2 and 3.1.3 explain the volume procedures. The mass of the volume fluid is known from the previous step. Its change is calculated from the difference of inlet

and outlet mass flow of the component (W_{B1} and W_B). New mass will affect the pressure in the volume which is calculated by E3.1.9. $\Rightarrow p_{B1\,new}$, $p_B = p_{B1}$

The temperature of the fluid leaving the volume is set to the temperature of the fluid in the volume from the previous time step:

$T_{tB} = T_{t\ vol\ old}$

The new volume fluid temperature that is going to be used in the next step is acquired from E3.1.10.

Turbine power:

$$TP = \eta_m \cdot W_A \cdot \left[h(T_A) - h(T_{tB1})\right]$$

New shaft rotational speed:

Initially, the turbine surplus power is evaluated:

$$SP = TP - CP - EP$$

Rotational speed increment is a function of turbine surplus power, the rotor inertia and the rotor rotational speed:

$$\frac{dN}{dt} = \frac{SP}{4\pi^2 I\ PCN_{old} \cdot N_{DP}}$$

The updated value of the relative rotational speed is obtained from explicit Euler integration:

$$PCN_{new} = \frac{1}{N_{DP}}\left(PCN_{old} \cdot N_{DP} + \frac{dN}{dt} \cdot \Delta t\right)$$

Convergent nozzle with fixed area

Inputs:

- p_{tA} Total pressure at the inlet
- T_{tA} Total temperature at the inlet
- A_B Nozzle exit area
- p_0 Ambient static pressure

Outputs:

- W_A Inlet mass flow
- W_B Outlet mass flow
- p_{sB} Static pressure at the outlet
- T_{sB} Static temperature at the outlet

Critical static temperature and pressure at nozzle exit:

The temperature is achieved if the nozzle is choked and the exit Mach number equals unity:

$$h(T_{cr}) = h(T_{tA}) + \frac{\gamma R T_{cr}}{2}$$

For the critical static pressure applies the same as for the temperature. From E2.4.8:

$$p_{cr} = \frac{p_{tA}}{e^{\left(\frac{1}{R}\int_{T_{cr}}^{T_{tA}} \frac{cp}{T} dt\right)}} = \frac{p_{tA}}{e^{\left\{\frac{1}{R}\cdot[\Phi(T_{tA}) - \Phi(T_{cr})]\right\}}}$$

The conditions in which the nozzle operates are determined according to the nozzle critical pressure ratio and the pressure ratio of nozzle inlet total pressure and ambient pressure ratio:

- The nozzle is unchoked if $\left(\frac{p_{tA}}{p_0} < \frac{p_{tA}}{p_{cr}}\right)$
- The nozzle is choked if $\left(\frac{p_{tA}}{p_0} \geq \frac{p_{tA}}{p_{cr}}\right)$

Static temperature at the outlet:

- For an unchoked nozzle: $p_{sB} = p_0$
- For a choked nozzle: $p_{sB} = p_{cr}$

Static temperature at the outlet:

- For an unchoked nozzle the temperature is obtained iteratively from the equation:

$$\Phi(T_{tA}) - \Phi(T_{sB}) = R \cdot \ln \frac{p_{tA}}{p_0}$$

- For a choked nozzle: $T_{sB} = T_{cr}$

The inlet and outlet mass flow:

$$W_A = W_B = \rho_B V_B A_B$$

Where:

$$V_B = \sqrt{h(T_{tA}) - h(T_{sB})}$$

$$\rho_B = \frac{p_{sB}}{RT_{sB}}$$

NOTE: Parameters that have not a direct impact on transient performance simulation (e.g. map scaling factors) have been deliberately left out from the example for the sake of simplicity)

3.2.4 Method Peculiarities

The previous chapter has shown the whole process of one transient step of a two spool gas turbine engine. The process is applicable also for engines with one- and three-spool architecture. For one-spool engine only one shaft speed is taken into account, whereas for a three spool engines there are total three shafts and every time step three rotational speed increments have to be calculated. The method assumes that all parameters, describing the component thermodynamic state, are calculated downstream the engine, apart from the mass flow, that is calculated upstream. The value of the mass flow in upstream components is therefore always several steps behind the mass flow of the components, where the change already took place (figure 3.8). For example the fuel flow in the combustor increases it has an immediate effect on combustor outlet temperature. In the subsequent turbine the increased temperature causes the inlet mass flow to change. It does not affect the turbine inlet

pressure, because this thermodynamic property is only calculated at the outlet. Therefore, any change in turbine inlet pressure ratio must be incited by an upstream component, more precisely, by control volumes. The information about the change in turbine inlet mass flow propagates upstream the engine, by a rate one component per transient step. Consequently, some engine bricks parameters are always updated several steps behind the change occurs. In chapter 3.2.5 an engine model transient study results are presented showing the consequences of lagging mass flow on engine parameters variation.

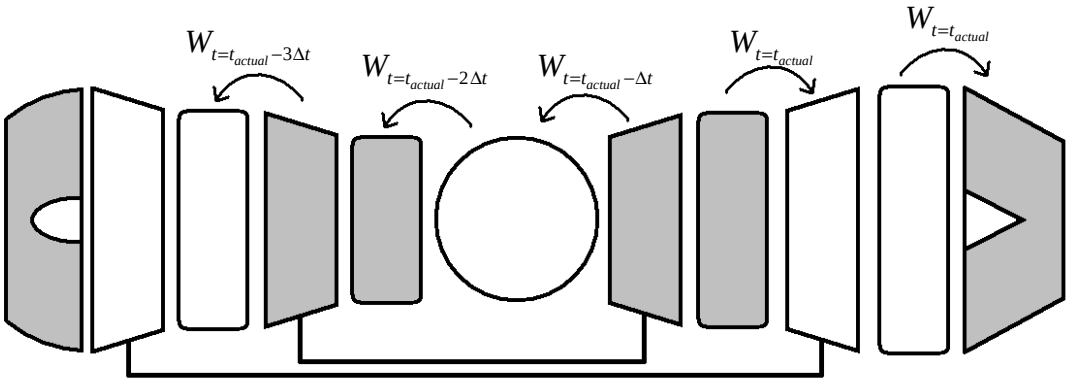


Figure 3.8: Image showing how the information of a change in mass flow after the combustor propagates through the engine.

There is one more complication about the mass flow being calculated upstream the engine. The resolution of this problem is discussed in the chapter 3.2.6

3.2.5 The Algorithm Test on Single Spool Turbojet

The Turbomatch version, in which the Rapid transient performance method has been implemented, did not yet undergo the source code clean-up (chapter 2.3). Engine model that could be tested was restricted to a simpler configuration.

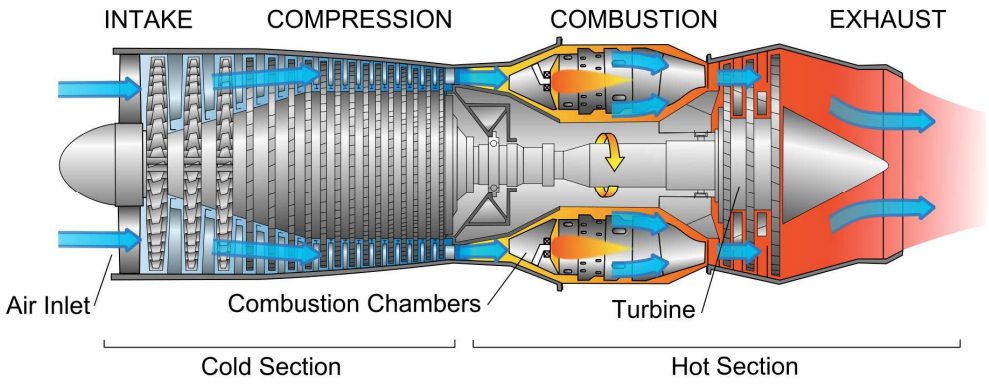


Figure 3.9: A cross-section of a turbojet (external source)

A model of a hypothetical single-spool turbojet has been generated (figure 3.10). It possesses the basic gas turbine components. The key engine specifications are featured in Table 3.1 and the design point ambient conditions in Table 3.2.

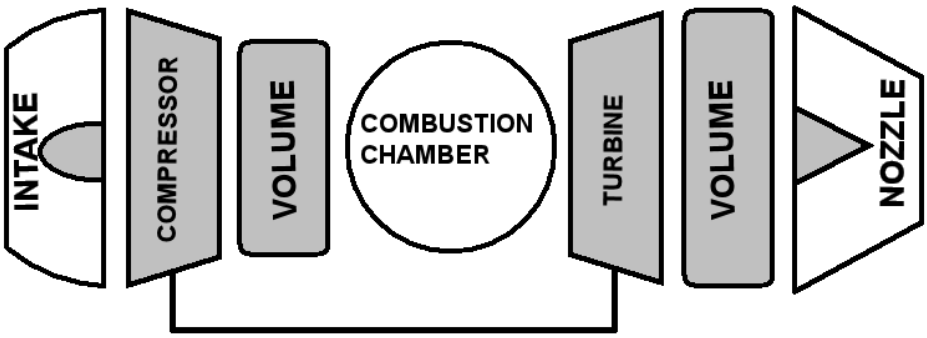


Figure 3.10: Turbojet testing model schema.

Parameter	Unit	Value
Inlet DP mass flow	kg/s	80.0
Compressor DP pressure ratio	-	10.0
Compressor DP isentropic efficiency	%	86.0
Compressor - combustor volume	m^3	0.1 – 1.5
Combustor DP fuel flow	kg/s	1.7797
Turbine DP isentropic efficiency	%	88
Nozzle duct volume	m^3	0.3
Rotor DP rotational speed	Hz	200
Rotor moment of inertia	$kg.m^2$	60.0

Table 3.1: Turbojet engine model specification

Parameter	Unit	Value
Altitude	m	133
Deviation from ISA temperature	K	5.5
Inlet Mach number	-	0.3
Air relative humidity	%	70

Table 3.2: Ambient conditions of the design point during the study

Simulation was run with three different values of compressor volume (comprising the volume of the compressor and the turbine) to demonstrate the effect of volumes on the simulation stability. The simulated duration of transient was 50 seconds. The computational time of engine models (table 3.3) imply that the method is faster than Thermodynamic matching transient method (table 3.4), but the computational time benefit decreases as the time step Δt decreases. For simple engine models involving only a few variables the simulation is calculated in the *real time* as this is the case. For more complex engines the computational time required would exceed the simulation time, thus the benefit of having the real time transient model would be lost. Much of the computational time is lost in many internal iteration loops used to calculate implicit equations of thermodynamic parameters within components. If these loops were replaced by equations in explicit form, where the effect of thermodynamic parameters on gas properties would be neglected, the simulation would be real time for any engine model with a consequent loss of accuracy.

Compressor volume [m^3]	Time step [s]	Duration of transients [s]	Computational time [s]
0.1	0.005	50.0	8.6
0.1	0.001	50.0	41.5
0.5	0.005	50.0	8.5
1.5	0.005	50.0	8.5

Table 3.3: Computational time of transient simulation

At the transient time t = 0 s the fuel flow has been sharply increased from W_{ff} = 1.2 kg/s to W_{ff} = 1.4 kg/s (figure 3.11). The following curves capture the engine dynamic response of the engine model.

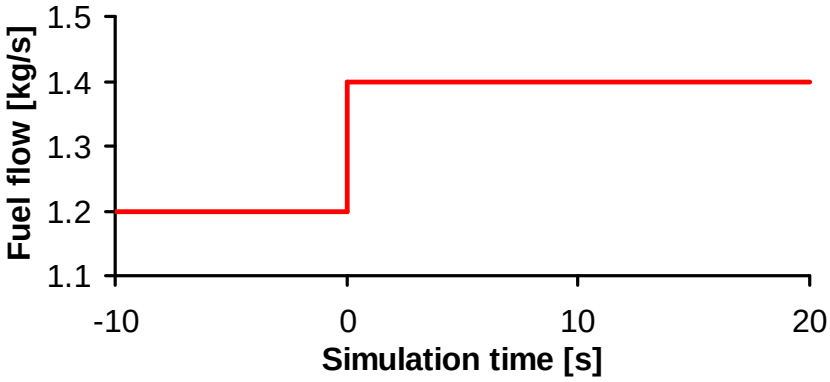


Figure 3.11: Step increase of fuel flow during the simulation

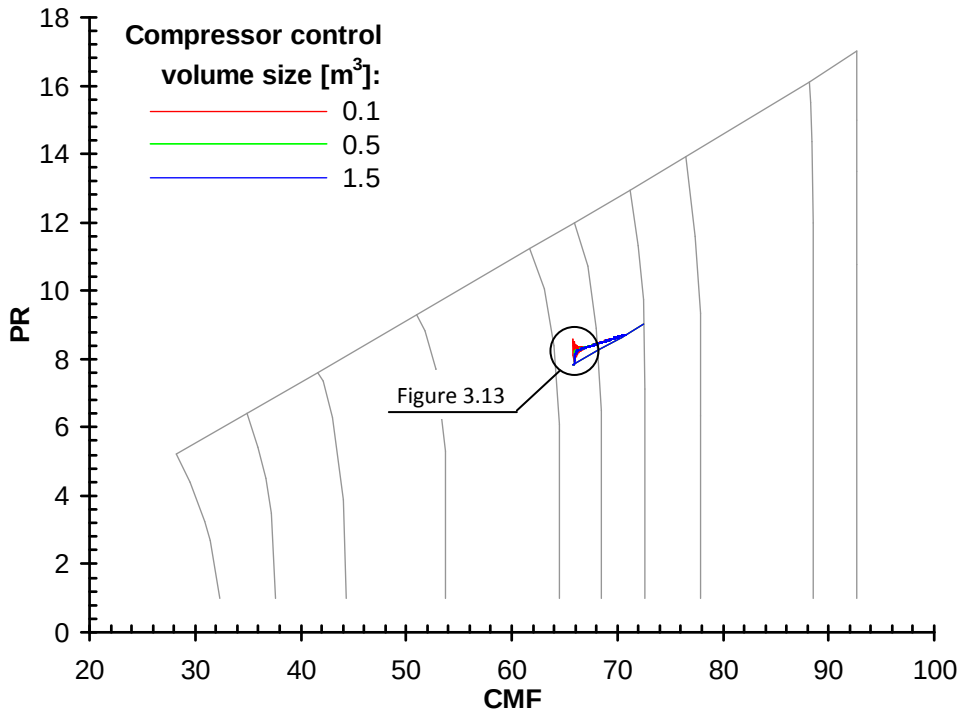


Figure 3.12: Compressor map with the running line of steady state and transient operation

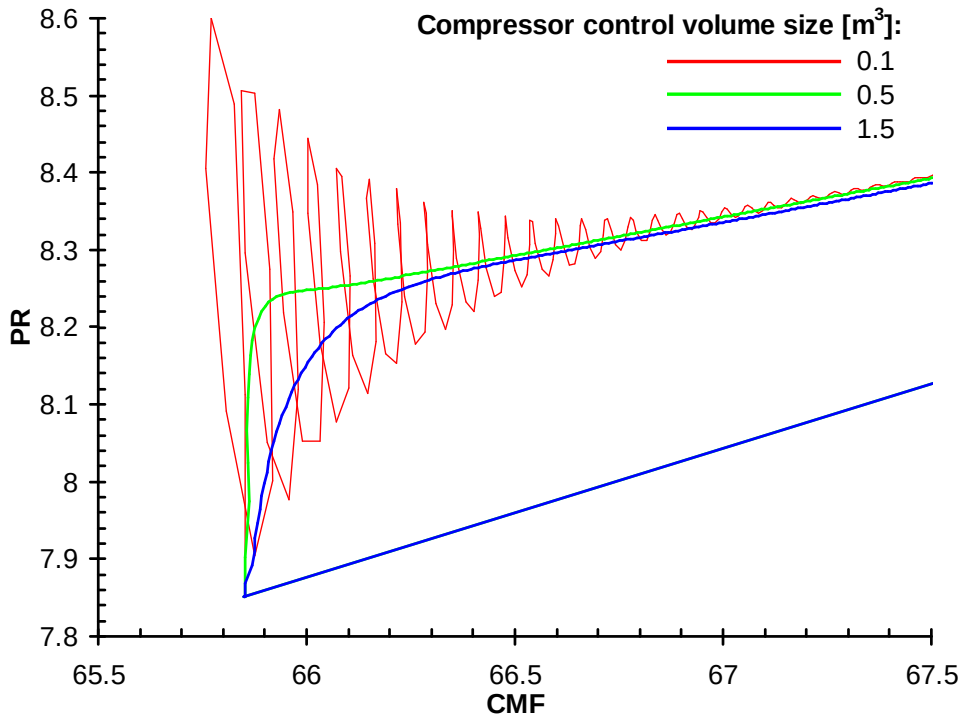


Figure 3.13: The detail of the initial transient period. The bottom-most line represents the steady state engine operation

Figures 3.12 and 3.13 show compressor running lines during steady state and transient operation. They display the dynamic response of the identical engine model with three different values of control volumes placed after the compressor. For the green and the blue curve the transient response is continual. The red curve on the other hand, shows the substantial oscillation followed by gradual stabilization of compressor pressure ratio, occurred due to the time lag between compressor outlet mass flow change and the turbine inlet mass flow change (chapter 3.2.4). A shorter time step will reduce the model instability but it will increase the computational time of the simulation.

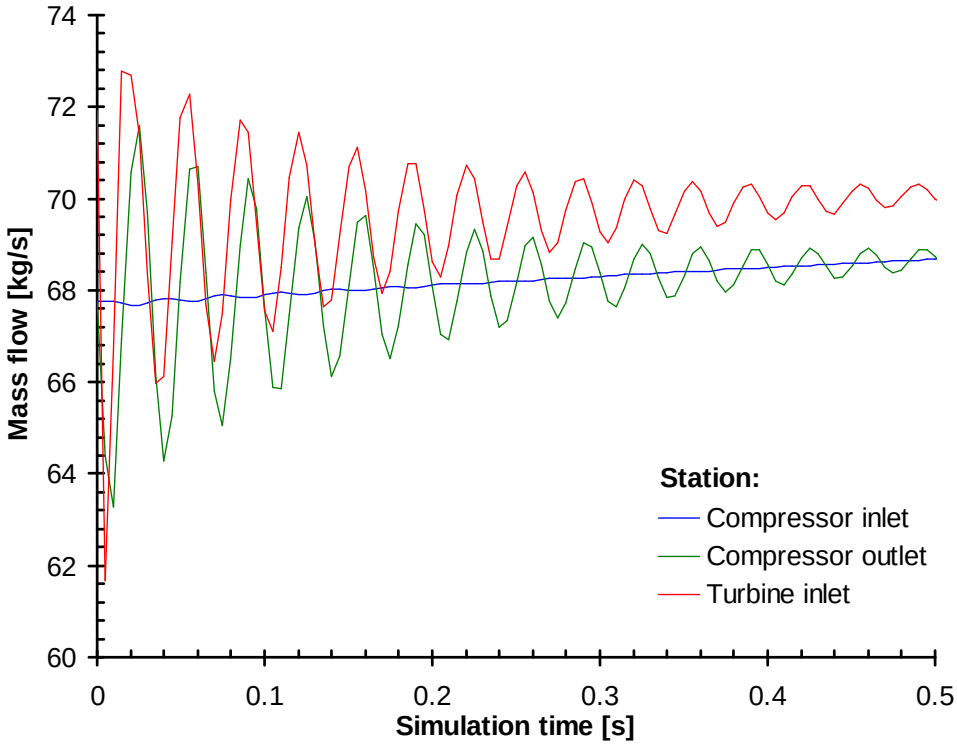


Figure 3.14: The variation of mass flow during transient simulation. Engine model compressor volume = 0.1 m^3

The mass flow curves on figure 3.14 reveal how compressor outlet mass flow is shifted few time steps behind the turbine inlet mass flow. The turbine mass flow is higher compared to compressor mass flow by fuel flow. The change in compressor outlet mass flow has the primary effect of changing the mass of fluid in the compressor control volume, thus it directly affects the compressor pressure ratio (figure 3.15). The change in inlet mass flow is predominantly driven by the change in relative rotational speed, which is for this case continual. Hence, the blue curve in the chart representing the compressor inlet mass flow, oscillates much less, compared to compressor exit and turbine inlet mass flow and its rise follows the rise of rotational speed. Although the simulation of engines with higher volumes seems to be without any oscillations, a closer observation of the very initial part of the simulation reveals small disturbance (appendix 3.1). The disturbances would reduce if smaller time step is chosen.

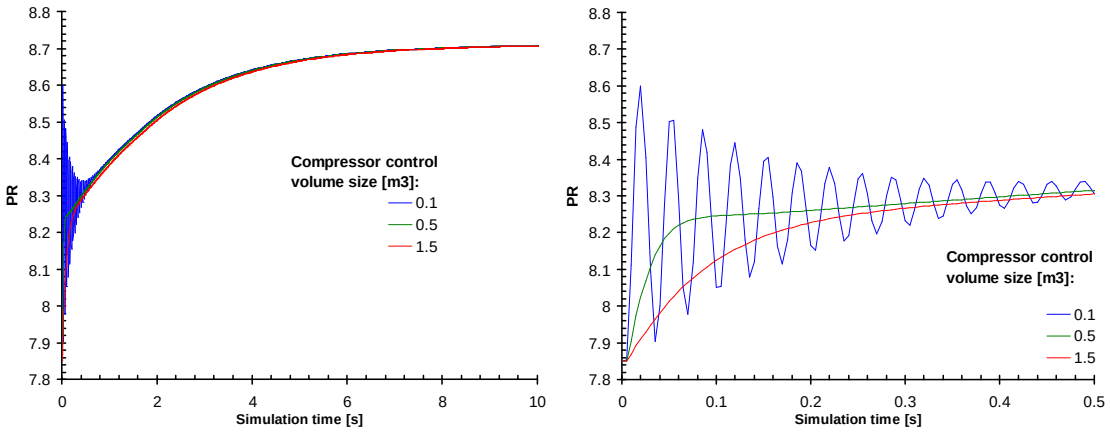


Figure 3.15: Compressor pressure ratio variation during transient performance simulation

The simulation tests also explain how a reduced time step can significantly reduce the initial oscillation in transient parameters (figure 3.16). A small disturbance at the beginning of simulation is unavoidable, but it is significantly reduced (figure 3.17).

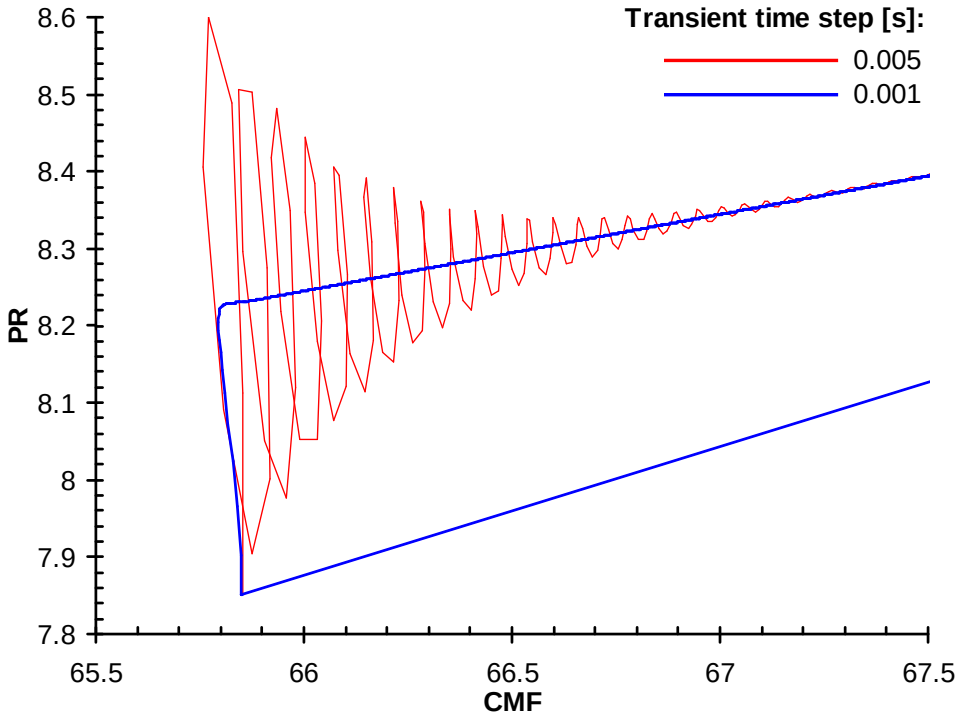


Figure 3.16: The stabilization of initial transient period for a smaller time step. Engine model compressor volume = 0.1 m^3

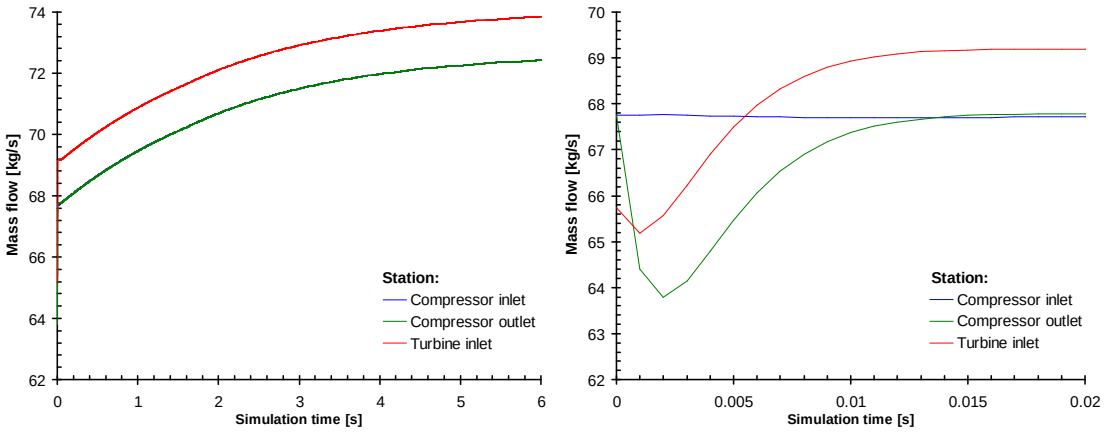


Figure 3.17: The mass flow variation during initial stage of transient simulation. Compressor volume = 0.1m^3, Δt = 0.001 s

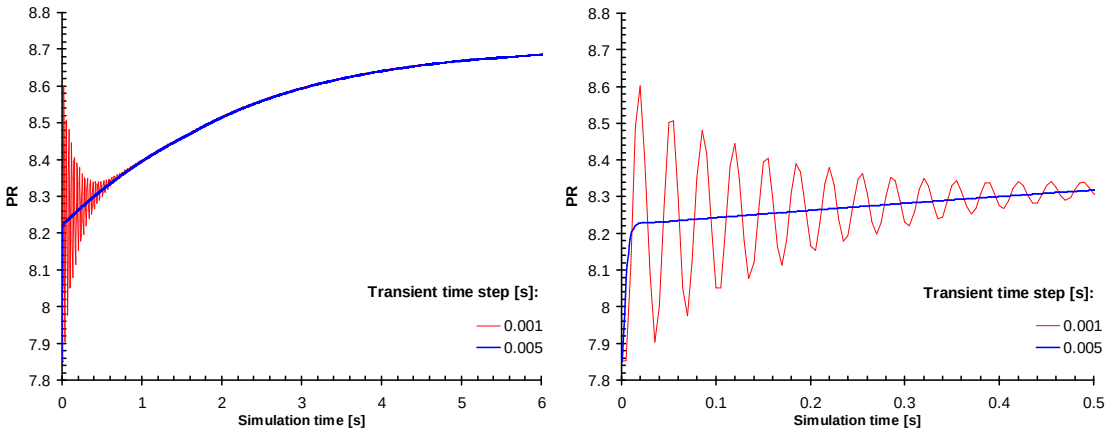


Figure 3.18: The effect of smaller time step on transient simulation stability, Compressor volume = 0.1m^3

3.2.6 Flow Segment

If the engine model involves one or more components where the flow is split into two streams (figure 3.19), such as bleed or bypass duct, the amount of fluid flowing through core and bypass passage is given by:

$$W_{out1} = \lambda_W \cdot W_{in} + \Delta W \qquad \text{E3.2.2}$$

where λ_W is the mass flow split ratio the ΔW is the additional mass flow that is extracted from the flow (negative value).

This component is in Turbomatch represented by a brick called PREMAS (figure 3.19). The two split streams may merge again after few components and if one stream has significantly lower mass flow than the core stream (e.g. mixing of core stream with cooling air in a turbine) the resulting flow after the mixing point is given only by the sum of the two flows (MIXEES in Turbomatch). For a mixing point where also the total pressure change resulting from a momentum balance is to be taken into account, the brick MIXFUL has to be used.

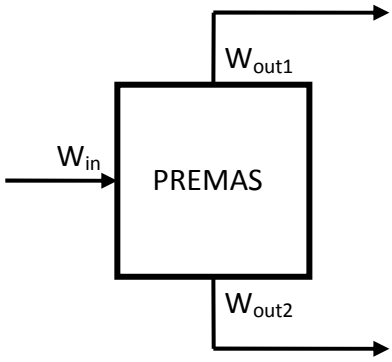


Figure 3.19: The schema and designation of PREMAS brick

For the Rapid transient performance model this flow segmentation constitutes a considerable complication, because the mass flow is not calculated downstream, as in the case of steady state calculation, but upstream. That means that for transient simulation the Turbomatch brick PREMAS will only evaluate the value of its inlet mass flow from mass flows of outlet 1 and outlet 2. In brick MIXEES however, the values of two inlet mass flows need to be determined by a certain ratio, that is not easy to calculate from the split ratio λ_W and mass flow loss ΔW of the corresponding PREMAS, because in-between these two components several other components may be involved in change in mass flows, such as combustion chamber (mass flow increases by fuel flow), the cold side of the heat-exchanger (mass flow leakage may occur), other MIXEES and PREMAS bricks, etc. An algorithm has been created that generates a dynamic neural network of all flows using pointer variables. When a variable is declared in the computer as a pointer, it does not have an allocated space of the

computer memory. It is generally created to deputize other variable. Therefore, if a pointer *P* is paired with other variable *A*, the expression:

P = P + 1

has the same effect as

A = A + 1

where a unity will be added to variable A.

There is also a possibility to allocate the pointer a memory space. In Fortran this is performed by statement:

```
allocate (pointer)
```

Pointer then behaves as a regular variable until deallocated. But the most interesting feature of pointer is that it can also point on itself. The only restriction is that pointer must be of the same type as the variable that it points to. Thus an integer pointer cannot point on a real-type variable or real-type pointer. All these features enable to create an own data-type pointer that behaves like a neuron – it stores certain variables and it points to other pointers of the same type. In Turbomatch a pointer variable *FlowSegment* has been created which was declared as own data-type:

```
type NodeType

    ! Indexing
    integer :: BrickCount
    integer,pointer :: IBR
    integer,pointer :: NNUMB
    integer,pointer :: Inlet1Num,Inlet2Num     ! Station number of
        ! first inlet (eventually also second inlet)
    integer,pointer :: Outlet1Num,Outlet2Num      ! Station number of
        ! first outlet (eventually also second outlet)
```

```
    ! Referencing
    type(NodeType),pointer :: PreBrick,PostBrick
    type(NodeType),pointer :: Inlet1,Inlet2
    type(NodeType),pointer :: Outlet1,Outlet2

    ! Values
    real,pointer :: LambdaW
    real,pointer :: DeltaW
    real :: k1,k2
    real :: m1,m2

end type NodeType

integer,target :: NoConnection = -1 ! If any of Inlet1Num, Inlet2Num,
    ! Outlet1Num, Outlet2Num have its value it means that they are not
    ! used
logical :: ContinueFlowPath

type(NodeType),pointer :: FlowSegment,PremierFlowSegment
```

The own data-type *NodeType* contains all necessary information to define flow path because at any brick the program is able to track all brick values of λ_W and ΔW of all components on the same flow path using the coefficients k_1, k_2, m_1 and m_2 that are calculated as follows:

For the first component in the engine model (usually the intake) the k_1 and m_1 coefficients are set to:

$$k_{Outlet1} = 1$$
$$m_{Outlet2} = 0$$

The coefficients referring to the second outlet from the component are not evaluated because intake only uses one outlet. For the compressor, combustor, turbine, duct pipe and the cold side of a heat-exchanger only the $m_{Outlet1}$ can be affected by the loss in mass flow. The original values of k and m was calculated in the preceding component. No splitting takes place in these components:

$$k_{Outlet1} = k$$
$$m_{Outlet1} = m + \Delta W$$

For the hot side of a heat-exchanger the mass-flow leaked from the cold side is added:

$$k_{Outlet1} = k$$
$$m_{Outlet1} = m - \Delta W$$

The PREMAS brick has two outlets and the coefficients have to be determined for following bricks on both of them:

$$k_{Outlet1} = \lambda_W \cdot k$$
$$m_{Outlet1} = \lambda_W \cdot m + \Delta W$$
$$k_{Outlet2} = (1 - \lambda_W) \cdot k$$
$$m_{Outlet2} = (1 - \lambda_W) \cdot m - \Delta W$$

As now every component stores the information about the value of the proportion of the mass flow flowing through this component and the overall mass flow entering the engine, for a mixing bricks MIXEES and MIXFUL (Figure 3.20) it is possible to calculate the proportion of mass flow entering the mixer to the total mass flow leaving it.

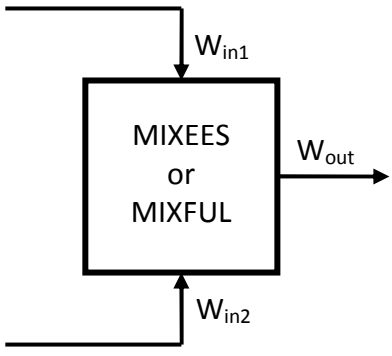


Figure 3.20: The schema of MIXEES (MIXFUL) mass flow relation

The mass flows W_{in1} and W_{in2} entering the mixing component is given by:

$$W_{in1} = \frac{W_{out}}{1+\frac{k_2}{k_1}} + \frac{k_2 m_1}{k_1} - m_2 \qquad \text{E3.2.3}$$

$$W_{in2} = W_{out} - W_{in1} \qquad \text{E3.2.4}$$

The algorithm allowing the calculation of the amount of mass flow entering the mixing component has been theoretically tested on a fictional flow model (Figure 3.21).

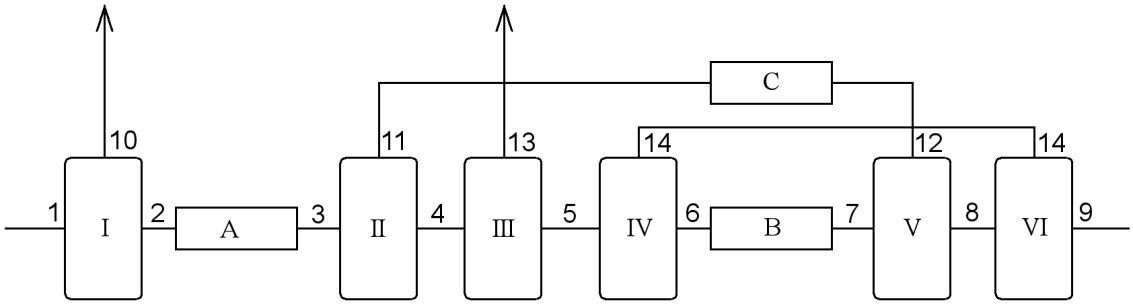


Figure 3.21: The mass flow schema of a hypothetical engine component layout

The model contains four components that split the mass flow and two mixing components which are denoted with Roman letters. Bricks A, B and C represent components where mass flow is only lost (or added from outside). Arabic numbers represent the stations of the model. The point is to show how E3.2.3 and E3.2.4 will be obtained by progressive calculation of mass flow leaving particular bricks:

Brick I:

$$W_2 = \langle (1-\lambda_I) \rangle_k \cdot W_1 \langle -\Delta W_I \rangle_m$$

$$W_{10} = \lambda_I \cdot W_1 + \Delta W_I$$

Brick A:

$$W_3 = W_2 + \Delta W_A = \langle (1-\lambda_I) \rangle_k \cdot W_1 + \langle -\Delta W_I + \Delta W_A \rangle_m$$

Brick II:

$$W_4 = (1-\lambda_{II}) \cdot W_2 - \Delta W_{II} = \langle (1-\lambda_{II}) \cdot (1-\lambda_I) \rangle_k \cdot W_1 + \langle (1-\lambda_{II}) \cdot (-\Delta W_I + \Delta W_A) - \Delta W_{II} \rangle_m$$

$$W_4 = \lambda_{II} \cdot W_2 + \Delta W_{II} = \langle \lambda_{II} \cdot (1-\lambda_I) \rangle_k \cdot W_1 + \langle \lambda_{II} \cdot (-\Delta W_I + \Delta W_A) + \Delta W_{II} \rangle_m$$

.

.

Gradually all components split ratios λ and mass flow losses ΔW will be encompassed in the mass flow equation for every station. The inlet mass flows of the first mixing component will be expressed by:

$$W_7 = W_6 - \Delta W_B = \langle (1-\lambda_{IV}) \cdot (1-\lambda_{III}) \cdot (1-\lambda_{II}) \cdot (1-\lambda_I) \rangle_{k1} \cdot W_I$$
$$+ \langle (1-\lambda_{IV}) \{ (1-\lambda_{III}) [(1-\lambda_{II}) (-\Delta W_I + \Delta W_A) - \Delta W_{II}] - \Delta W_{III} \} + \Delta W_{IV} - \Delta W_B \rangle_{m1}$$

$$W_{12} = W_{11} + \Delta W_C = \langle \lambda_{II} \cdot (1-\lambda_I) \rangle_{k2} \cdot W_1 + \langle \lambda_{II} \cdot (-\Delta W_I + \Delta W_A) + \Delta W_{II} + \Delta W_C \rangle_{m2}$$

After the substitution of k and m constants:

$$W_7 = k_1 \cdot W_1 + m_1$$

$$W_{12} = W_8 - W_7 = k_2 \cdot W_1 + m_2$$

and several basic arithmetic operations the sought correlations E3.2.3 and E3.2.4 are obtained.

The second algorithm implemented into Turbomatch is based on the thermodynamic matching method. The way in which a solution is obtained is similar to steady state off-design, where all component parameters at the inlet are either known or guessed and the parameters at the outlet of the component are calculated. The number of guesses must equal the number of errors – constraints which have to be satisfied if a solution is found. The major difference with the transient method is that the power produced by the turbine does not equal the power consumed by the driven set of the compressor. If engine components also contain control volumes, its effect on thermodynamic parameters is taken into account as well. The main advantage of the method is its accuracy. At every transient step all thermodynamic parameters have updated values (unlike the rapid transient performance method, where some thermodynamic parameters have obsolete values due to the process by which mass flow is evaluated) and the calculation moves to the next time step only if all engine constraints are satisfied. Therefore, such parameter oscillation as shown in figure 3.13 will not occur in

engine models analyzed with this method. Instead the transient curves are always smooth and only a negligible noise at the outlet parameters is caused by the matrix iteration, where the solution picks up random values from a defined tolerance region. Although this method is considered slower in terms of computational time due to the matrix iteration, in some cases it permits a longer time step due to its lower propensity to parameter oscillation.

The foundations of the method are discussed in (Walsh, Philip 2004; Pilidis 2006) and (NATO 2007) and they emanate from principles introduced in (Fawke, Saravanamuttoo 1971) and have been applied in (GSP 2004). The foundation of the method implemented into Turbomatch follows the approach suggested in these references, although it has been slightly adapted to the program structure.

3.3 Thermodynamic Matching Transient Method Turbomatch Adaptation

The process of thermodynamic matching transient simulation is shown in figure 3.22. It begins with steady state calculation that produces the initial values for thermodynamic parameters at all stations of engine model. After that the program mode switches to transient and the simulation time is set to initial value.

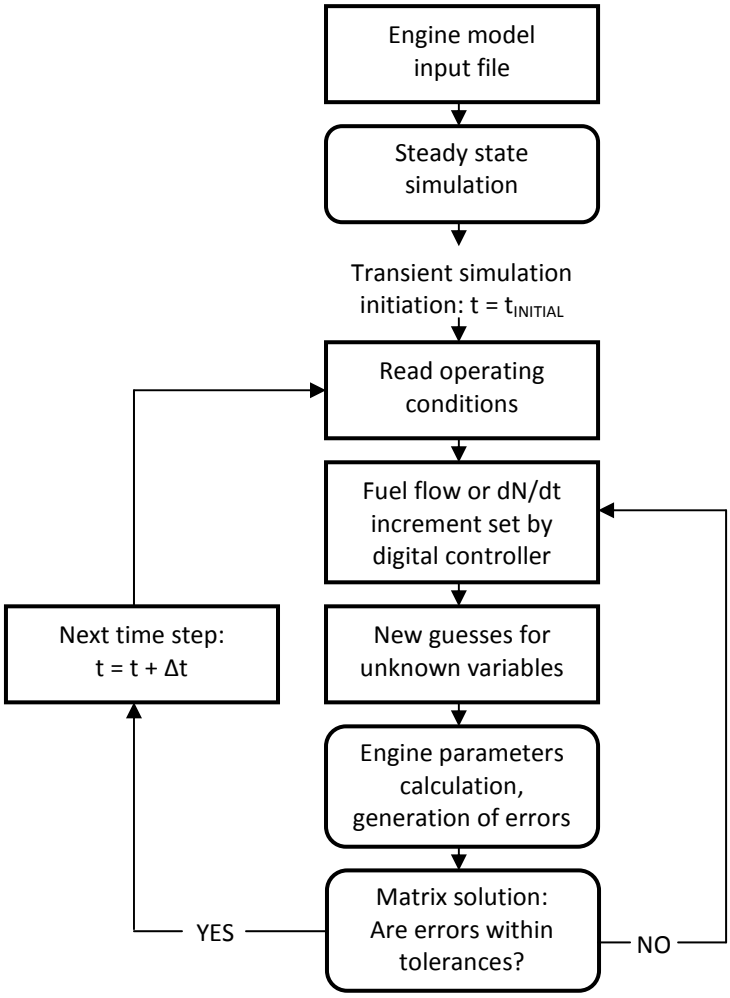


Figure 3.22 Thermodynamic matching transient model flow chart

The operating conditions are read from the external transient input file (chapter 4.3). Changes in operating conditions are listed against the engine simulation time. It is hence possible to simulate the dynamic response of a change of any parameter that can be controlled during steady state simulation. Fuel flow is also read from the file, either as a precise value or in terms of fuel flow schedule where referred fuel flow is a function of referred rotational speed in a look-up table. Subroutine *BurnerFuelControl()* uses the look-up table determine the actual fuel flow for the analyzed time step (chapter 4.3.3).

The iterative matching of transient step calculation starts with the guess of new matching variables. In the original method (Walsh, Philip 2004) the addition of a new variable for a transient iteration is proposed: For every shaft the surplus power (expressed as the difference between the turbine power delivered and power consumed by all compressors driven by that turbine) should be guessed. A new variable requires a new constraint to be matched, the calculated fuel flow needed to produce the guessed surplus power must equal to fuel flow from the control system. If the engine possesses more than one shaft, apart from the fuel flow the inlet corrected mass flows of all turbines downstream the HP turbine have to be matched.

In Turbomatch however, there is in fact one excess variable used in steady state analysis. For a jet engine configuration two variables are guessed in the compressor (the surge margin Z and the relative rotational speed PCN) and one in the turbine (the corrected mass flow at the inlet). All available variables and matching constraints are found in appendix 2.10. Since the value of the rotational speed PCN is known from the previous step an attempt was initially made to guess the value of the new rotational speed for which the required surplus power would be calculated:

$$\left(\frac{dN}{dt}\right)_{now} = \frac{\left(PCN_{now} - PCN_{last}\right)\cdot N_{DP}}{\Delta t}$$

From which the surplus power is

$$SP = 4\pi^2 I\ N \cdot \left(\frac{dN}{dt}\right)_{now}$$

This approach is theoretically faultless, but it would require more advanced NLE solver. The Jacobian matrix (E2.3.2) is created numerically in the current solver, which means a fractional value is added to variable to produce the derivative for the matrix:

$$\frac{df_i}{dx_j} = \frac{f_i(x_j + \varepsilon) - f_i(x_j)}{\varepsilon} = \frac{f_i(x_j + \Delta x_j) - f_i(x_j)}{\Delta x_j} \qquad \text{E3.3.1}$$

The increment to the original variable is in solver calculated as:

$$\Delta x_j = VarCoef \cdot x_j \qquad \text{E3.3.2}$$

The coefficient VarCoef is set initially to

$$VarCoef = 0.001$$

The transient time step is usually very small (several milliseconds and less) and so the change in rotational speed per one time step is. The change in rotational speed calculated using VarCoef would give very high values of PCN_{new} resulting in high values of Surplus power needed to achieve such change in PCN. The value of turbine enthalpy difference dh, calculated from compressor power and surplus power, will have odds to be located far beyond the available turbine map. The problem could be solved if the value VarCoef was a function of transient time step and the compressor power.

A faster and more elegant solution is to replace the variable for transient simulation for surplus power. The subroutine *SwitchVariables()* localizes which variable corresponds to the PCN and replaces it with a new variable, the surplus power. The surplus power is not guessed directly for the similar reason as in case of guessing the value of new rotational speed. The problem only appears at the first time step of transient simulation that follows steady state calculation, because the value of surplus power in steady state mode is zero and to calculate the initial derivatives of NLE functions upon the surplus power for the Jacobian matrix, the numerical increment for the variable can not be determined from the variable value (E3.3.2), when it equals to zero. The increment must be determined by other approach which ensures that the calculated value of surplus power will be not too large, so that the operating point on the turbine would be not located outside of the map, and therewithal it also will be not too small. If it was smaller than the arithmetic operation precision (appendix 2.3), the sum of compressor power and surplus power will be equal to compressor power only. The surplus power is therefore guessed indirectly.

During every transient time step the value of compressor power from the previous step is saved in the value *ComWkOld*. Instead of utilizing the surplus power as a

variable, the value of fake turbine power is guessed. The surplus power for the actual step is thus determined as:

$$SP = TurWkVar - ComWkOld$$

This way, the NLE solver can easily calculate the variable increment using the VaeCoef coefficient. TurWkVar is denoted as fake turbine power because it does not equal to the value of turbine power, since the variable ComWkOld does not represent the compressor work for current time step.

The new value of the shaft rotational speed is evaluated from the guessed surplus power (E3.0.4, E3.0.6). The position of the compressor operating point on the map is determined by the relative rotational speed and the corrected inlet mass flow which means that all parameters at the compressor outlet (and the volume inlet) can be directly calculated.

The thermodynamic matching process during transient simulation resembles the off-design simulation matching (Chapter 2.4.4). Only this time the volume dynamics and heat storage in components is assumed and the PCN variable has been replaced by the surplus power. After the matching for one step is successful the subroutine *OUTSEQ()* plots the results of the converged time step if it has been externally set to do so, the simulation time is incremented by one time step and all values rotational speeds, masses in control volumes and temperatures of fluid in control volumes are stored as "old".

3.3.1 Volume Dynamics Variables

The thermodynamic matching transient algorithm is also based on intercomponent volume packing (Chapter 3.1.2) and heat storage (Chapter 3.1.3) principles. Here, the component outlet mass flow is not known from the downstream component as it was the case for the rapid transient method (Chapter 3.2.2). The outlet mass flow from the

volume component is therefore guessed as variable *WoutVar*. The matching constraint for the mass flow variable is the pressure in the volume that has to match with the pressure calculated at the outlet of the engine component with which the volume component is coupled. The outlet mass flow variables are not in use for the engine model working in steady state mode. They, and their pressure constraints are only activated subroutine *assignMassVars()* when transient simulation begins. A question might arise asking why the outlet mass flow from the volume component is not calculated from the volume pressure, when this value is known. The new mass of the fluid in the volume may be calculated from E3.1.9 and from the difference between volume masses of the current and the previous time step the outlet mass flow could be determined E3.1.8. At the initial stages of the algorithm development this approach has been chosen assuming that saving variables and constrains for volume dynamics will speed up the process significantly. When the method has been tested however, it was found that this approach is instable, leading to negative volume outlet mass flows for large pressure drops. The model is especially sensitive to this phenomenon when small volumes were used. The problem appears when the pressure increase in the volume is too large, giving a high mass flow at the outlet of the volume. If the downstream component has a map with mass flow parameter (such as compressor or turbine), its operating point is then off the map. Choosing a smaller time step would not solve the problem either, because then the effect of saving the computational time would be lost.

3.3.2 Fuel Control in Transient Operation

In modern gas turbine engines digital control systems are used. Their purpose is to ensure that the engine delivers required power, or thrust, so that the operating point will not reach unsafe regions. The control systems are computers that operate mechanisms which handle certain parameters. The computer monitors the engine thermodynamic parameters, picks up the data from the engine operator and after processing them it sends orders to control mechanism what action is required. The engine operator is either a computer, if the engine is controlled automatically or

human that operates the computer. For instance, the engine is fully operated by a computer in auto-pilot mode, but during take-off, landing and other maneuvers, requiring human control a person controls the computer. During a human control the input for the computer is usually the angle of the power lever. The most obvious parameter controlled by the engine control system is the fuel flow. But except for the fuel flow, depending on the engine complexity also other engine parameters may be controlled.

Parameters controlled by the engine control system:

- Combustor fuel flow
- Afterburner fuel flow
- Compressor VGV angle
- Turbine NGV angle
- Propelling nozzle throttle area
- Compressor air bleed

When these parameters are changed the transient response of the engine is invoked. The development of the transient performance engine model is therefore often accompanied by the development of the control system.

The transient model without any control system is only able to control the fuel flow and other engine control parameters according to the simulation time. Every change in fuel flow is then set externally. This approach is rather arduous for the user, as he has normally limited possibilities to find out about the variation of the fuel flow through the time. Therefore a subroutine *BurnerFuelControl()* was created which controls the amount of combustor fuel flow.

The first task of the subroutine simulating the control system is to evaluate the fuel flow according to the fuel flow schedule table where the referred fuel flow is listed against the referred rotational speed. This table is delivered externally in the transient

input file. The referred fuel flow is fixed in the steady state operation for different ambient conditions if referred rotational speed is also fixed (Walsh, Philip 2004). Increasing the value of referred fuel flow will cause the power imbalance and engine starts to accelerate. Similarly, if the referred fuel flow value is decreased the engine shaft decelerates. Shaft continues to accelerate until the value of referred fuel flow is set back to the steady state value. The acceleration rate is approximately constant if the referred fuel flow is increased by the same amount. If it is increased by the same percentage the acceleration rate tends to increase with increasing rotational speed. When the increase of the referred fuel flow is calculated several effects, which have an impact on engine shaft acceleration rate need to be taken into consideration. For example, for the same referred speed the referred fuel flow decreases with flight Mach number due to ram pressure increase at the intake and at lower shaft speed the effect of unchoked nozzle has also to be taken into consideration (figure 2.23).

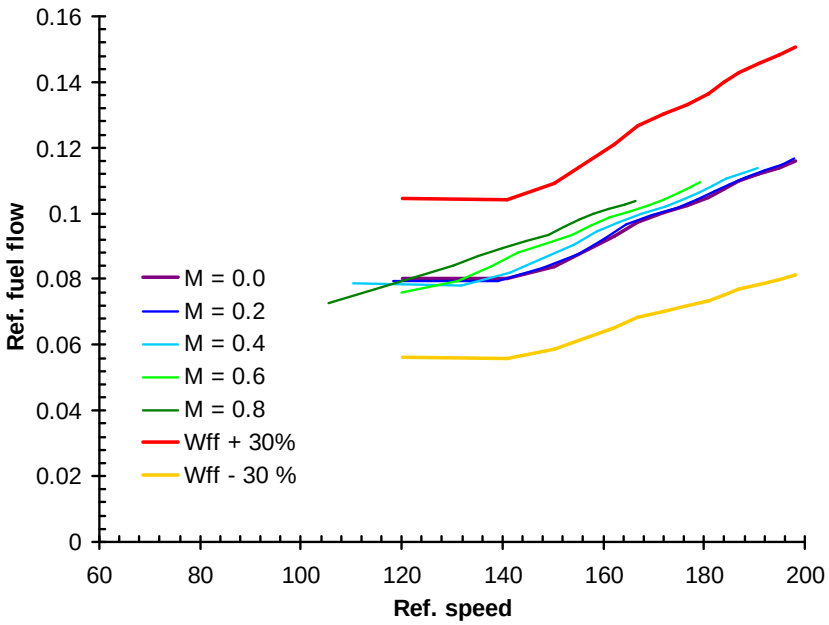


Figure 3.23 Referred fuel flow variation for different Mach numbers. Red and gold line represent the over-fuelling and under-fuelling respectively

The increase in operating altitude tends to increase acceleration times roughly by the ratio of altitude pressure to ISA SLS pressure. When referred fuel flow and rotational

speed are fixed the referred acceleration rate is fixed as well (Walsh, Philip 2004). Actual acceleration rate consequently increases (E3.3.3).

$$\frac{dN}{dt} = \left(\frac{dN}{dt}\right)_{referred} \cdot \frac{p_{ISA\,SLS}}{p_{compressor\,inlet}} \qquad \text{E3.3.3}$$

Bleed and power extraction, deterioration increase steady state referred fuel flow, which means that acceleration is performed slowlier and deceleration faster. The over-fuelling must be high/low enough to ensure the engine acceleration and deceleration at all conditions

The second task of the fuel schedule subroutine is to ensure that the fuel flow increase will not be abrupt, but spread evenly through a very short time period (figure 3.24).

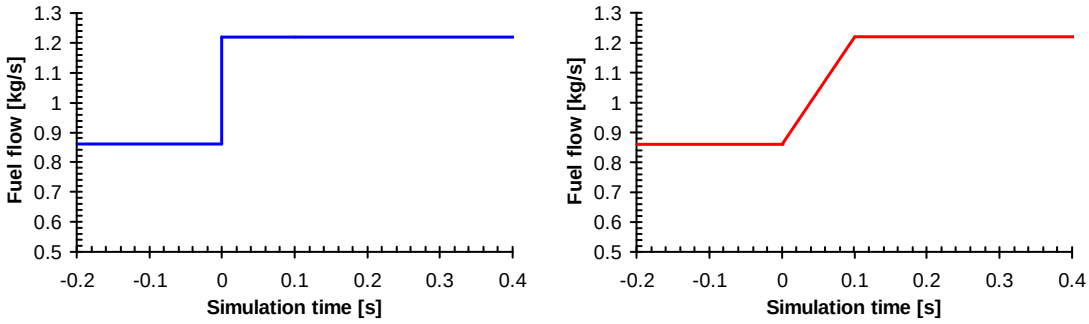


Figure 3.24: Chart showing abrupt fuel flow increase demand (left, blue curve) is spread over a time period (right, red curve)

Even in real engines if the fuel flow is set to increase by the control system, the change is not performed abruptly. The time between the fuel flow starts to increase and reaches the level set by the control system is called *transition time*. It usually takes few tenths of a second. In Turbomatch the abrupt change in a parameter could cause a failure of the solution convergence because the initial guesses may be far away from the roots, returning the operating point of the engine model calculated for an initial loop away from a possible working region. On the other hand, if the changes of parameters for the next step are small the solution lies closer to previous solution, which means that the calculation is performed faster. Therefore any change in fuel

flow is spread evenly through 1/10 of a second. Since the time step is often 0.01 or lower, the fuel flow increase per one step is only 1/10 of the required increase, or lower.

The third task of the program control system is to ensure that the operating point of components always stays within their map range and that no extinction in combustor occurs. The maximum and minimum value of the combustor outlet temperature can be externally specified. If the combustor outlet temperature exceeds the limiting value, the fuel flow schedule is no longer calculated from the fuel flow – speed table, but maintained at the same level until it drops back below the limit. Similarly, if the combustor outlet temperature drops below the minimum limiting value any further decrease in fuel flow is suspended until it returns back above the limit.

3.3.3 Transient Performance Test Studies

This subchapter discusses about the engine models which have been tested on the new developed program upgrade that involves the capability of transient performance simulation using the thermodynamic matching method. Engine models chosen for transient simulation tests can be found in chapter 2.3.8. They are proxy for configurations and architectures of major gas turbine engines used in the industry as well as in aviation. The models are identical to those used during program clean-up and debugging, though new parameters have been added which are required for transient study. The parameters include:

- The volumes of components
- The rotor inertias of each shaft
- The design point rotational speed of every shaft

Apart from design point shaft rotational speed which is usually known from engine documentation, the values for the transient engine parameters were assumed. For a rigorous transient analysis the parameters should be either known from an external

source, or calculated by a sizing algorithm. The purpose of test studies was to calculate dynamic responses of engine models from chapter 2.3.9 and prove that the program is stable for any engine configuration and it will converge to solution under all operating conditions for which the engine was designed. The analysis results of four engine models from the range of models presented in chapter 2.3.9 are presented here:

Engine model 1: Single spool turbojet

The design point and specifications of this engine model is thoroughly described in chapter 3.2.5 where the dynamic response on an abrupt change in fuel flow is investigated using a rapid transient performance method. The same dynamic response is simulated here using the second, thermodynamic matching transient method. Thus a comparison of benefits for the methods can be made. The configuration of the gas turbine engine is illustrated on figure 3.10. The engine specifications are featured in Table 3.1 and the ambient conditions for the design point study in Table 3.2.

Transient performance is undertaken under same conditions as in chapter 3.2.5 to study the effect of the compressor volumes and simulation time step. The simulation duration was 50 seconds. At the transient time t = 0 s the fuel flow is abruptly increased from W_{ff} = 1.2 kg/s to W_{ff} = 1.4 kg/s. For the sake of exactness the fuel flow control system has been turned off, thus the fuel flow schedule follows the same pattern as on figure 3.11. The time required for the simulation is approximately two times and 40 percent longer for 5 ms and 1 ms time step respectively (Table 3.4).

Compressor volume [m^3]	**Time step [s]**	**Duration of transients [s]**	**Computational time [s]**
0.1	0.005	50.0	17.6
0.1	0.001	50.0	57.1
0.5	0.005	50.0	17.2
1.5	0.005	50.0	17.1

Table 3.4: Computational time of transient performance simulation of single-spool turbojet using thermodynamic matching

Although the time to simulate the same engine model is longer, the method stability is better compared to rapid transient performance, because for every time step all parameters across the engine are matched, protected from oscillation problems seen on figure 3.13. Therefore a longer time step can be chosen in some cases saving the computational time considerably.

Figure 3.25 shows the working line of the compressor during steady state and transient performance. Figure 3.26 shows the detail of the working line at the initial part of transient performance caused by the sudden increase of combustor fuel flow, causing the combustor outlet temperature to rise immediately. The COT after the fuel flow rise is much higher compared to the steady state value, appertaining to the same fuel flow, because the air mass flow entering the combustion chamber is smaller. The fuel-to-air ratio starts to decrease after the combustor air mass flow increases. At the beginning of chapter 3.1 it is explained how the change in combustor outlet temperature causes the increase of compressor pressure ratio. Although the theory is based on an engine model without any volume dynamics and with the turbine exit choked, the essential outcome for the theory is valid for all gas turbines: The pressure ratio increases with the combustor outlet temperature. In the transient model the pressure ratio increase is evoked by the initial decrease of combustor mass flow. At the beginning of dynamic response the mass flow into compressor control volume stays unchanged, but the outlet mass flow decreases with combustor mass flow, causing the pressure in control volume to rise (figure 3.2). The bigger the volume the longer the pressure increase. Due to the excess turbine power the shaft rotational speed accrues, allowing higher mass flow in the turbine, the compressor and all other engine components. The curves on figure 3.26 illustrate this effect. The pressure ratio increase is steeper for engine models with smaller control volumes.

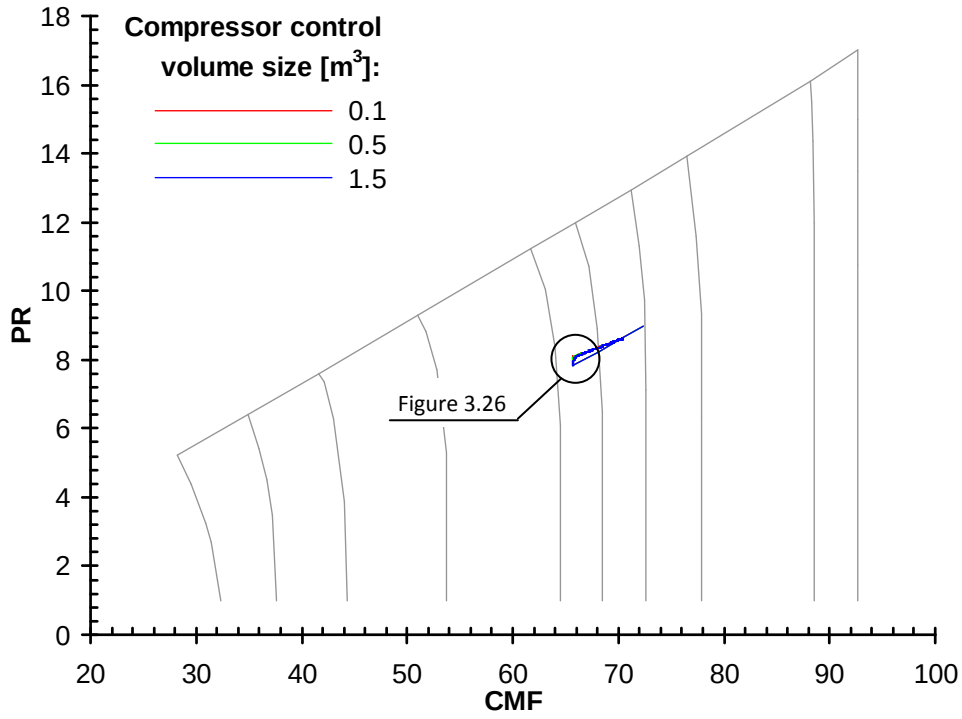


Figure 3.25: Turbojet working line for steady state and transient performance

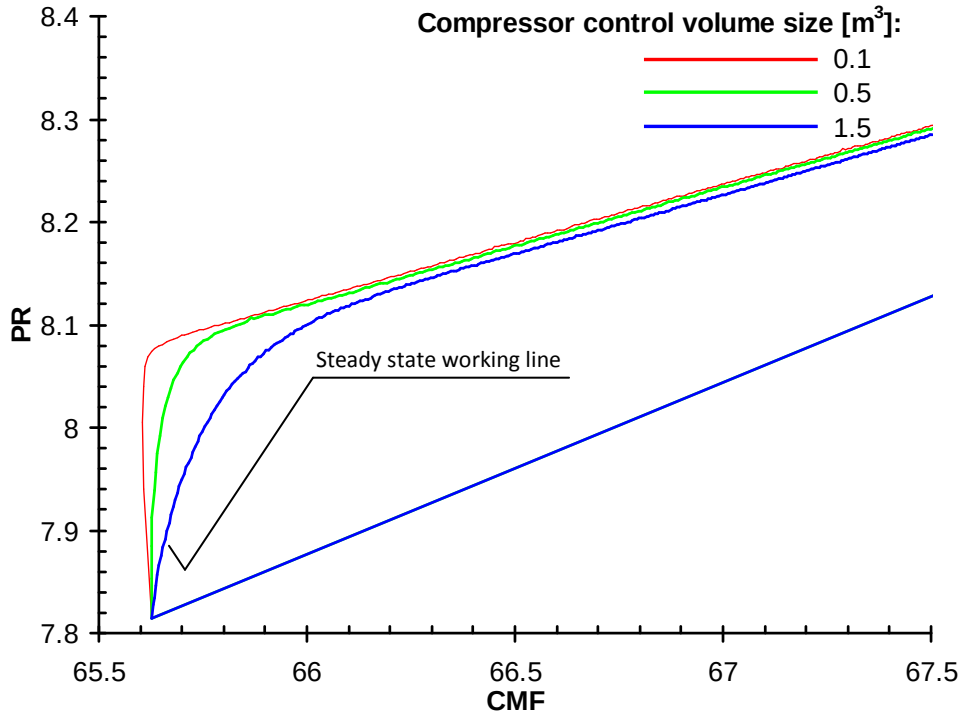


Figure 3.26: The detail of compressor working line for initial part of transient simulation

There is no lag between turbine inlet and compressor outlet mass flow change (figure 3.27) because all engine parameters are matched at every time step. The increase in compressor inlet mass flow is slower compared to the outlet mass flow, because it depends predominantly on the relative rotational speed, which increase is at every time step fractional. The compressor inlet mass flow dependence on the compressor pressure ratio is minimal, because of the gradient of the compressor map speed lines (figure 3.25).

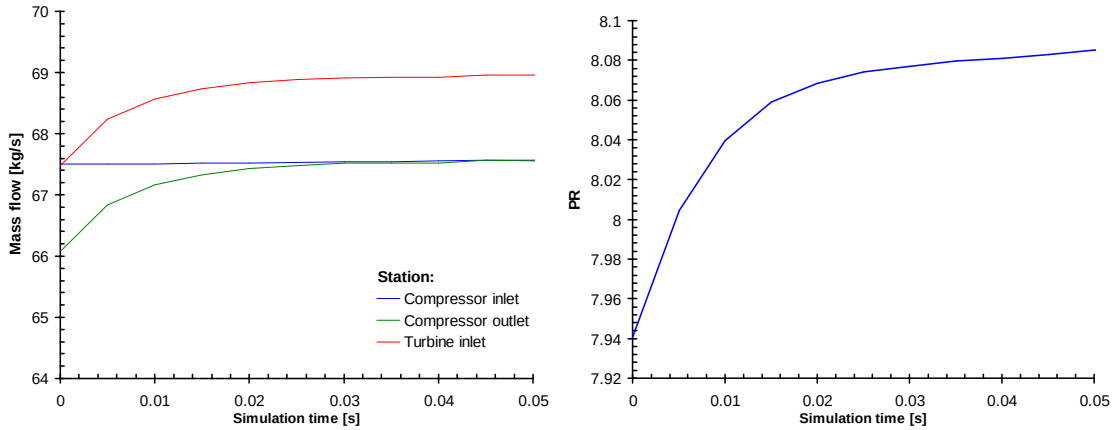


Figure 3.27: The variation of mass flow and pressure ratio during transient performance

Engine model 2: One spool gas generator with power turbine

The configuration of gas turbine engine with one spool and one gas generator (figure 3.28) resembles the single spool turbojet configurations from the first case study, whereas for this case the hot, pressurized exhaust gas is not discharged into ambient to produce the thrust, but instead it is expanded on the power turbine which delivers useful work.

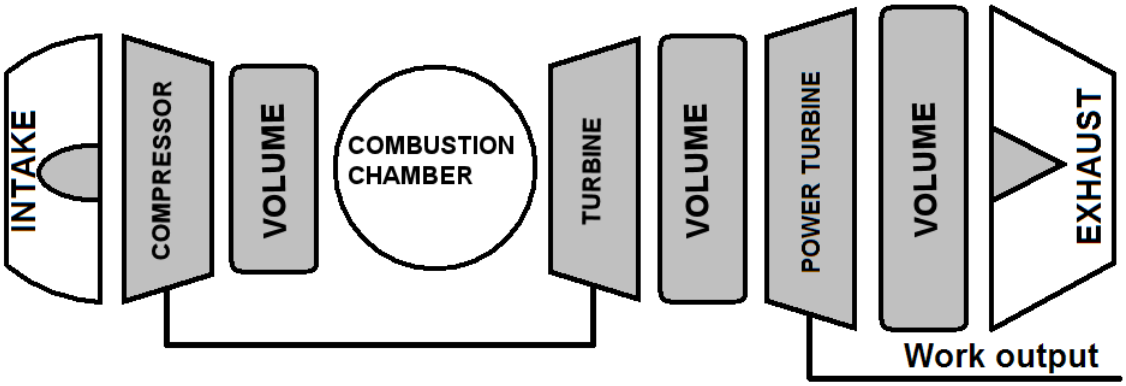


Figure 3.28: The configuration of the gas turbine engine model with free power turbine

The design engine specifications are listed in table 3.5. It is assumed that the power turbine utilizes all of the useful work from the gas leaving the compressor turbine, and the exhaust nozzle pressure ratio is set to 1.02. A pressure ratio is kept above the ideal value (which is unity) to simulate small losses in the exhaust nozzle. The sea-level static ambient conditions are used during the design point and transient performance simulations.

Parameter	Unit	Value
Inlet DP mass flow	kg/s	77.2
Compressor DP pressure ratio	-	8.8
Compressor DP isentropic efficiency	%	84.0
Compressor - combustor volume	m^3	0.05
Combustor DP fuel flow	kg/s	1.032
Compressor turbine DP isentropic efficiency	%	87
Compressor turbine volume	m^3	0.05
Compressor turbine rotational speed	Hz	200
Gas generator shaft moment of inertia	$kg.m^2$	30.0 – 50.0
Power turbine DP isentropic efficiency	%	87
Power turbine volume	m^3	0.15
Power turbine rotational speed	Hz	50
Power turbine shaft moment of inertia	$kg.m^2$	50.0

Table 3.5: Engine model specification

At the time t = 0 s the fuel flow is suddenly increased from 1.107 to 1.352 causing the combustor outlet temperature to rise immediately from 1175 K to 1318 K (figure 3.29). Gradually the engine mass flow increases reducing the FAR and consequently the COT.

The effect of rotor inertia on engine acceleration is studied. Engine with higher shaft inertia requires a longer time to achieve a new steady state (table 3.6 and figure 3.29). Table 3.6 also suggests that the increasing shaft inertia has little effect on computational time. It has also a very small effect on compressor working line (appendix A3.3.2), almost imperceptible for the range of shaft inertias studied.

Gas generator shaft inertia [kg.m^2]	Time step [s]	Time to SP ~ 0 [s]	Computational time of 10 seconds of transient [s]
30	0.001	5.7	30.59
40	0.001	7.7	30.95
50	0.001	9.5	30.37

Table 3.6: Computational time of transient simulation

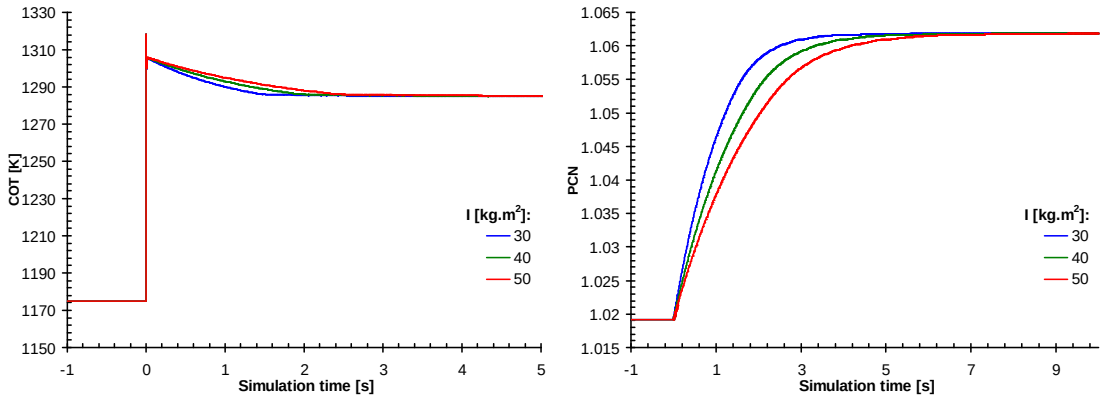


Figure 3.29: The variation of combustor outlet temperature (left) and relative rotational speed of gas generator shaft (right) during transient acceleration

Engine model 3: Two spool turbofan

The third engine model shown here has the configuration of a two spool turbofan (figure 3.30). The inlet mass flow separates into core and bypass stream after the first compressor (or fan). The core stream flows through the HP compressor, the combustion chamber and the HP turbine, and discharged on LP turbine which drives the fan.

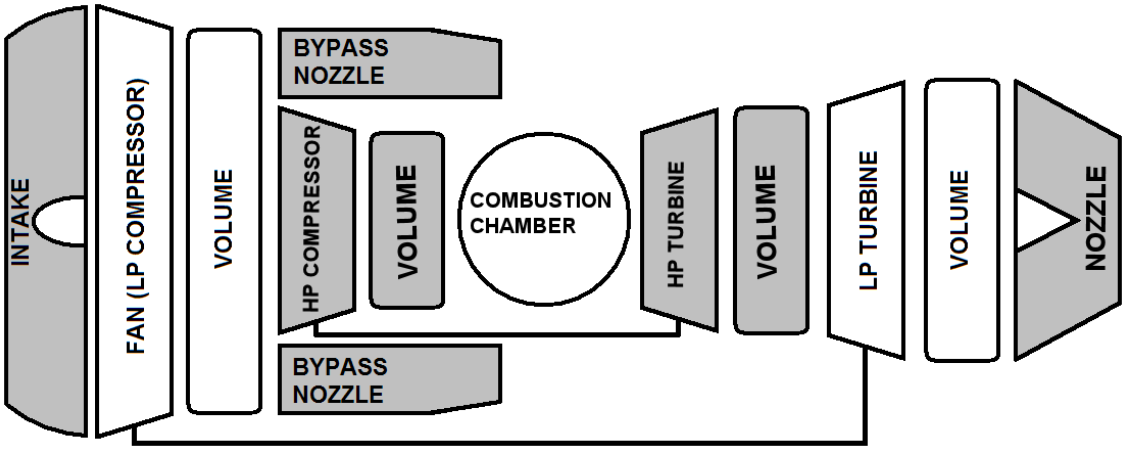


Figure 3.30: The schema of a two spool turbofan

The design point of the engine model was set to the top-of-climb with pressure altitude of 10 668 m (35 000 ft). The design point Mach number is 0.8, no humidity is assumed. The engine specifications (table 3.7) are stated for these hypothetical flight conditions.

Parameter	Unit	Value
Inlet DP mass flow	kg/s	150
Fan (LP Compressor) DP pressure ratio	-	1.8
Fan (LP Compressor) DP isentropic efficiency	%	89.0
Fan (LP Compressor) volume	m^3	0.4
Bypass ratio	-	4.9
HP Compressor DP pressure ratio	-	18.2
HP Compressor DP isentropic efficiency	%	86.0
HP Compressor – Combustor volume	m^3	0.1
HP turbine cooling flow	%	10.0
Combustor DP fuel flow	kg/s	0.4
HP turbine DP isentropic efficiency	%	92.0
HP turbine volume	m^3	0.09
HP turbine rotational speed	Hz	200
HP spool shaft moment of inertia	$kg.m^2$	55
LP turbine DP isentropic efficiency	%	91
LP turbine and nozzle duct volume	m^3	0.3
LP turbine rotational speed	Hz	100
LP turbine shaft moment of inertia	$kg.m^2$	70

Table 3.7: Two-spool turbofan model specification

The transient performance is not performed at the top-of-climb, but at the ground level under static conditions. Three types of transients undertaken with three different fuel schedules were simulated. For the step increase the fuel flow was abruptly increased to the final value. Acceleration modes I and II use the control system to calculate the fuel flow from the look-up table of referred fuel flow versus referred HP rotational speed (figure 3.31). For the acceleration mode I, the referred fuel flow values used for the acceleration is increased by 30% from the steady state values. The mode II assumes that the acceleration referred fuel flow is increased from the steady state referred fuel flow by a constant value. The computational time increase due to the employment of the fuel flow control system is negligible (table 3.8)

Transient case	Time step [s]	Time to SP ~ 0 [s]	Computational time of 10 seconds of transient [s]
Step increase	0.0005	4.53	58.14
Acceleration I	0.0005	3.98	59.92
Acceleration II	0.0005	3.99	61.61

Table 3.8: Computational time of transient simulation

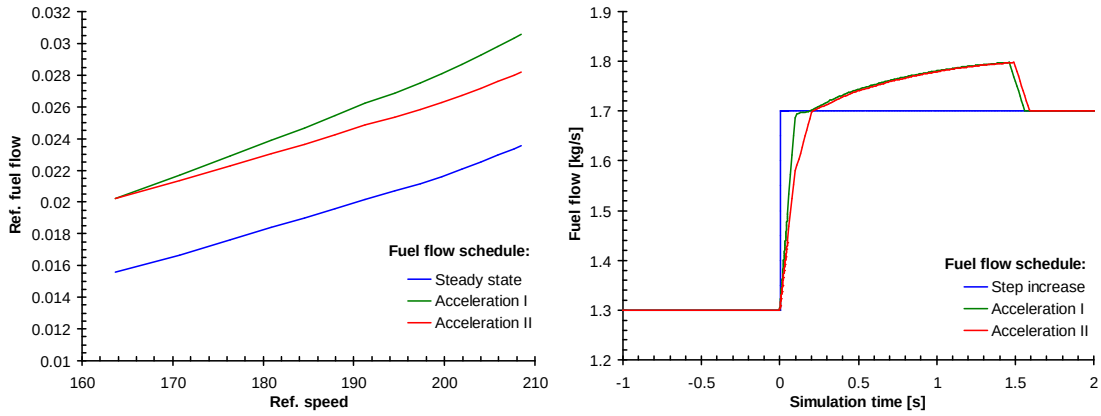


Figure 3.31: Transient fuel flow schedule – the control system fuel schedule function (left) and the fuel flow variation with time (right)

The maximum permissible combustor outlet temperature is limited to 1890 K when the control system is used. Control system thus does not only ensures that the highest gas temperatures will not exceed critical values (figure 3.32), which depend on the

combustor and turbine material melting point and cooling effectiveness, but it also decreases the time needed to reach new steady state given by HP rotational speed (table 3.8, appendix A3.4.2). Although one would anticipate the acceleration in mode I will be evidently faster than for the mode II because of higher overfueling, it does not happen for this case. During most of the time of the acceleration the fuel flow is calculated from the COT limit and not from the referred fuel flow function, therefore the acceleration time saving at the acceleration mode I is minimal.

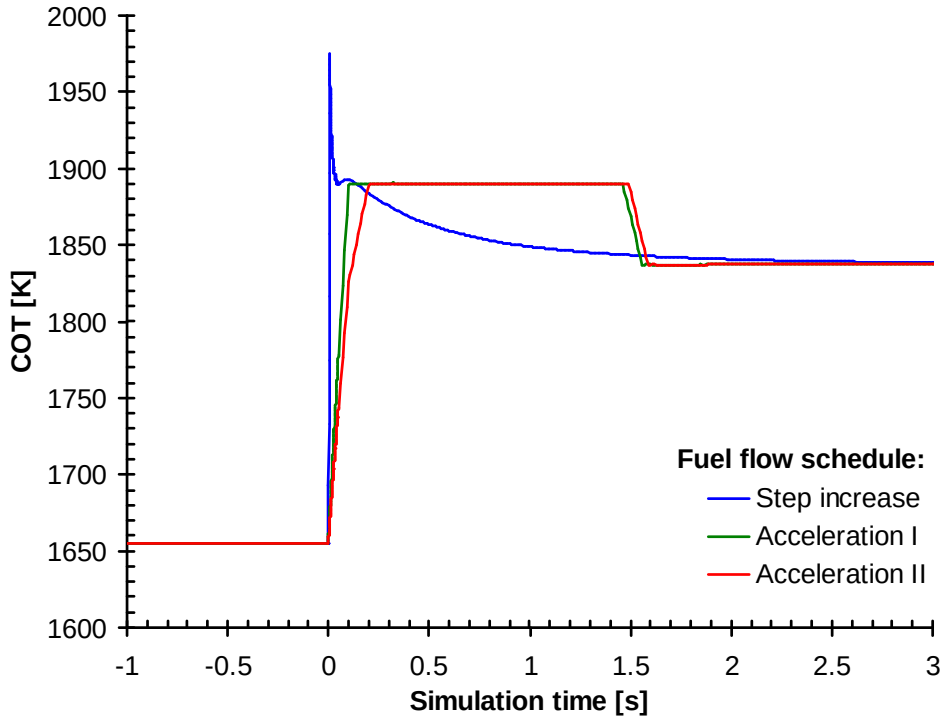


Figure 3.32: The variation of combustor outlet temperature during transient acceleration

The transient working line of the HP compressor (appendix A3.4.1) starts with the sudden increase of the pressure ratio. For the step increase in fuel flow the transient working line gradually approaches the steady state line until it reaches it. For the controlled transient acceleration the steep increase of the pressure ratio approaches the steady state running line at lower rate than for step increase. When the desired HP rotational speed is achieved the control system suddenly cuts the fuel flow to the

steady state value and the transient working line consequently falls until it reaches the steady state compressor working line. The transient working line of the LP compressor almost blends with the steady state working line. Because the bypass mass flow is even for ground conditions approximately five times bigger than core mass flow, the shape of the transient net thrust curve resembles the curve of fan rotational speed. The fan rotational speed is at some point of the transient operation higher than the final value because of the effect of volumes

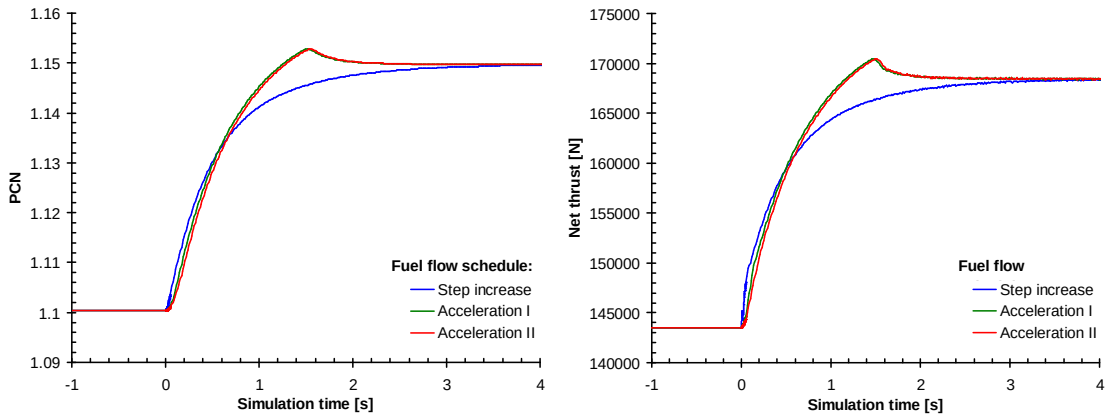


Figure 3.33: The variation of fan relative rotational speed (left) and the overall engine net thrust during transient performance

Engine model 3: Three-spool turbofan

The last and the most complex engine model on which the transient performance simulation is demonstrated is the three-spool large-bypass turbofan with six control volumes involved (figure 3.34).

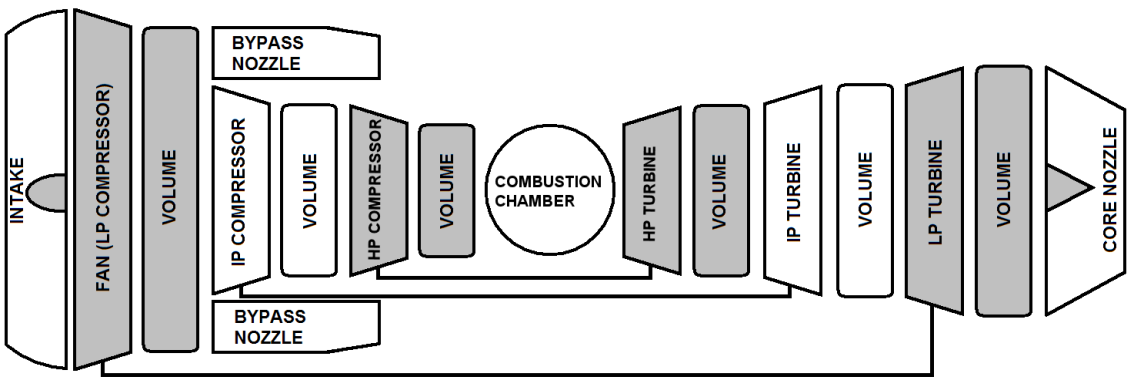


Figure 3.34: The schema of three-spool turbofan engine model

The design ambient conditions correspond to ISA SLS with ambient temperature increased by 15°C. Design engine specification are found in table 3.9. This time the design combustor outlet temperature was specified instead of fuel flow. Even during off-desing simulation, which is run to produce steady state fuel flow function (figure 3.35) for the control system, COT is used as a handle. During transient operation the control system handles the amount of fuel flow, not the combustor outlet temperature.

Parameter	Unit	Value
Inlet DP mass flow	kg/s	1179
Fan (LP Compressor) DP pressure ratio	-	1.56
Fan (LP Compressor) DP isentropic efficiency	%	87.4
Fan (LP Compressor) volume	m^3	1.5
Bypass ratio	-	8.5
IP Compressor DP pressure ratio	-	5.188
IP Compressor DP isentropic efficiency	%	84.6
IP Compressor – Combustor volume	m^3	0.5
HP Compressor DP pressure ratio	-	5.188
HP Compressor DP isentropic efficiency	%	85.1
HP Compressor – Combustor volume	m^3	0.9
HP turbine cooling flow	%	10.0
Combustor outlet temperature	K	1800.0
HP turbine DP isentropic efficiency	%	88.5
HP turbine volume	m^3	0.2
HP turbine rotational speed	Hz	200
HP spool shaft moment of inertia	$kg.m^2$	150
IP turbine DP isentropic efficiency	%	90.9
IP turbine and nozzle duct volume	m^3	0.4
IP turbine rotational speed	Hz	160
IP turbine shaft moment of inertia	$kg.m^2$	180
LP turbine DP isentropic efficiency	%	91.5
LP turbine and nozzle duct volume	m^3	0.7
LP turbine rotational speed	Hz	100
LP turbine shaft moment of inertia	$kg.m^2$	280

Table 3.9: Two-spool turbofan model specification

The demonstration shows the engine transient behavior for an increase in COT from 1400 K to 1800 K. The fuel flow function for the control system assumes 10%

overfuelling. Limiting COT value was 1890 K, but because of slow acceleration the limit has never been reached so the fuel flow is only a function of the fuel flow function input into control system (figure 3.35). Because of the engine's complexity and very small time step the computational time needed to simulate 60 seconds of engine operation exceeds eight minutes (Acceleration I in table 3.10). The computational time could be significantly reduced if volumes are excluded from the engine model (Acceleration II). All results presented here refer to the engine model with the control volumes.

Transient case	Time step [s]	Time to SP ~ 0 [s]	Computational time of 60 seconds of transient [s]
Acceleration I	0.0005	25.9	523.27
Acceleration II	0.002	25.9	82.86

Table 3.10: Computational time of transient simulation

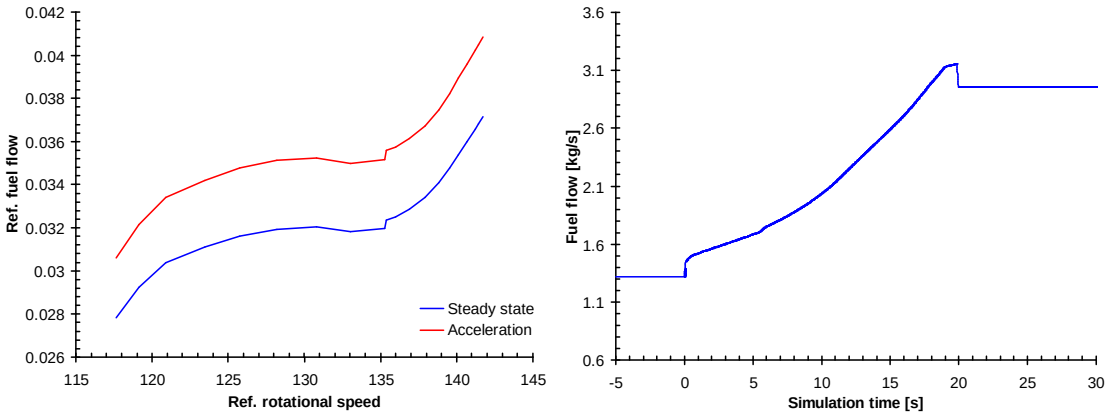


Figure 3.35: The fuel flow function for the control system (left) and the fuel schedule during transient performance simulation (right)

Appendix 3.5 shows the compressor working lines during the transient acceleration. The working line of the HP compressor is the most distant from the steady state line, whereas the working line for the LP compressor almost coincides with the steady state. When the combustor outlet temperature is increased at the beginning of transients, the HP turbine inlet mass flow (and the outlet mass flow of the HP compressor control volume) decreases causing the pressure ratio in the HP compressor to rise. But the HP

compressor inlet mass flow changes at much slower rate. This is caused by the shape and the position of the compressor map speed lines. Thus the outlet mass flow from the IP compressor control volume and the IP compressor pressure ratio are not immediately increased with the fuel flow, but they rise gradually as the IP spool speeds up. The IP compressor working line is hence much closer to steady state line then HP compressor working line. A similar behavior applies to the LP compressor. Its pressure ratio increase rate is the smallest among all compressor because it has to "wait" for all succeeding compressors' rotational speeds to rise.

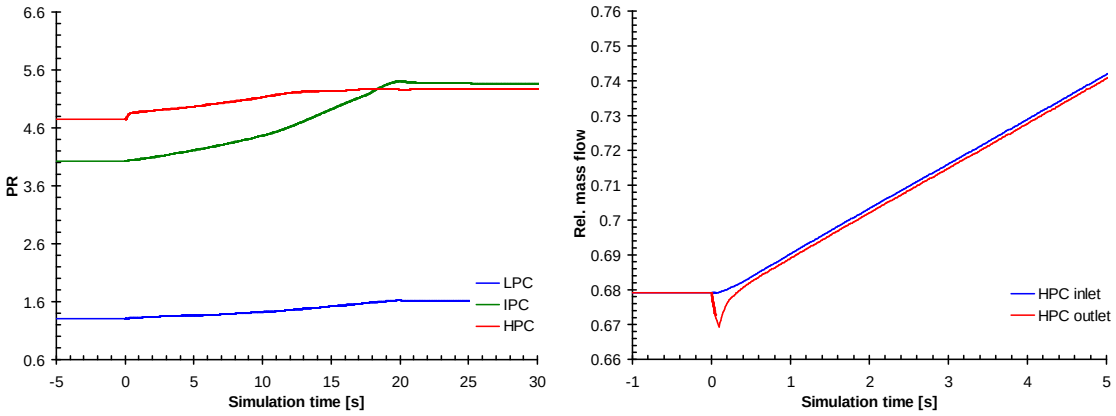


Figure 3.36: The variation of compressor pressure ratios (left) and the relative HP compressor mass flow (right) during transient performance

The engine net thrust during the transient performance is predominantly a function of fan rotational speed because of large bypass ratio. Because the fan transient working line almost coincides with the steady state line, the fan mass flow and pressure ratio increase gradually with rotational speed (figure 3.37).

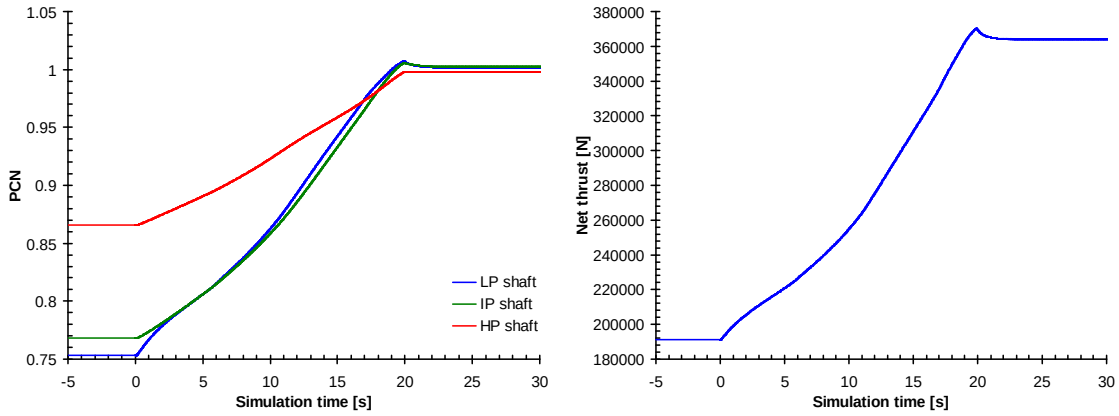


Figure 3.37: The variation of shaft relative rotational speeds (left) and overall engine net thrust (right) during transient performance

3.4 Conclusion

Two promising transient methods have been adapted and implemented into Turbomatch including the volume packing and heat storage effect.

The rapid transient performance method has been implemented as first. The rotational speed change was calculated using the explicit Euler integration. The method does not require iterative solution for engine model variables and every transient time step is calculated by the single engine loop calculation. In this method the component pressures and temperatures are calculated downstream the engine model, whereas the mass flow is calculated upstream. That means that for every component the value of outlet mass flow is known and the inlet mass flow needs to be calculated. For an engine model where flows are split (bypass, turbine cooling air extraction, customer bleeds, etc...) an algorithm has been developed, which allows calculating the ratio of fluid flow in the tow stream dynamically at the location where the flows merge. The method has been tested on a single spool turbojet where results showed oscillations of several thermodynamic parameters at the initial part of transients. The oscillations were reduced when smaller simulation time step was chosen. The method would probably be favored if the simulation was carried out in the real time or faster, however it works in real time only for simple engine configuration. Many routines used

in components use iterative algorithms anyway, increasing the computational time significantly

The second implemented method, which uses the thermodynamic matching algorithm, showed greater stability in convergence and results. Moreover it is less sensitive to time steps. In some cases larger simulation time steps may be chosen than in the rapid transient performance method causing the simulation to run faster. The rotational speed for each step is calculated from the evaluated compressor-turbine power imbalance. The original rotational speed variable is in transient simulation mode replaced by other variable – the turbine surplus power, expressed as the difference of turbine power and compressor power. The upgraded program has been tested on 11 engine models with different configuration. The results of transient behavior of four of them has been presented and analyzed.

Because of several new parameters that are needed for transient performance simulation, the input and output engine model files underwent several modifications:

- A new transient input file was introduced. It must have the same name as the engine model file with the extension *.ttr or other, chosen in Turbomatch user preference file. The file is not requested for steady state analysis. It contains data that are only needed for transient simulation such as fuel schedule, brick data and station vectors that user wishes to change during transient, time of the simulation, step length, etc... Next chapter explains more detail about the transient input file.
- Because of high number of transient steps, results for transient are not plot into *.tmr file. Results are only available in tabular *.4 file formatted and they should be post processed for a better view.
- The user has the possibility to switch the volume dynamics routines ON or OFF, if faster simulation is required and lower accuracy of the simulation is acceptable. If all volumes are switched off the program will use constant mass flow method to calculate the transients.

- Console time output table has been included to inform the user of actual engine runtime test.
- Fuel control has been implemented (Fuel function table must be provided) with an option to input different fuel schedules for different times along with the max/min temperature limitation for combustor outlet temperature.

The implemented transient algorithm gives user a freedom to change and control all parameters which are allowed to change during steady state simulation. Various transient cases (acceleration, deceleration, different fuel schedule, changing Mach number and altitude, effect of firing a missile, reslam, etc.) can be thus simulated with the upgraded program.

CHAPTER 4

THE TURBOMATCH QUICKSTART

The description of installation process, the engine model input file and a guide how to run a transient performance simulation is presented in this chapter. The whole process of generating an engine model input file and specifying transient conditions is explained here. The Quick Start guide begins with program installation procedure, and then explains the details of engine model input file of a single-spool turbojet.

4.1 Program Installation

Turbomatch has a console user interface which means the user can only communicate with the program using the console window using text files which are read by the program or by typing characters by keyboard when the program asks to do so. Current version is only compiled for Windows NT (or higher). Two files and one component maps folder are required for successful execution of the program:

maps_standard contains tabbed data of 5 different compressor characteristics, 6 turbine characteristics, one burner map and one heat-exchanger map. Details on the maps can be found in Turbomatch manual. This folder can be placed into arbitrary folder (e.q. C:\Turbomatch\)

Turbomatch Main.exe is the software executable file. It can be run on any computer in Microsoft Windows operating system. The file can also be placed into arbitrary folder (e.q. C:\Turbomatch\)

TURBOMATCHpreferences.txt Contains user settings for Turbomatch. This file must be placed only into one folder found by one of following methods:

1. Run *Command Promt* (on the bottom pane of Windows click start -> Run and type “cmd” without quotations mark. After the > prompt, type in “echo %HOMEDRIVE%%HOMEPATH%” without quotation marks. Windows system responds with a folder name – this is the first place where TURBOMATCHpreferences.txt can be placed
2. Run *Command Promt* (on the bottom pane of Windows click start->Run and type “cmd” without quotations mark. After the > prompt, type in “echo %USERPROFILE%” without quotation marks. Windows system responds with another folder name (but it may be the same) – this is the second place where TURBOMATCHpreferences.txt can be placed

Turbomatch preference file can be accessed via any suitable text editor as MS Notepad, Notepad++ (not MS Word!) and overwritten. The contents of the file may look as follows:

```
!TURBOMATCH preferences file

TM$ForOneInputFileGo YES
TM$ManualInput NO

TM$InputFolder C:\Documents and Settings\User\Desktop\TM Folder\TM Input

!TM$InputFolder F:\TMtest
TM$OutputFolder C:\Documents and Settings\User\Desktop\TM Folder\TM Output

TM$StandardMapsFolder D:\Turbomatch\maps_standard
!TM$AdditionalMapsFolder D:\Turbomatch\maps_additional

TM$OutputResultsFile *_2results.txt
TM$OutputPlottingFile *_3outputs.3
TM$OutputTabularFile *_4tabular.4
TM$TransientInputFile *.ttr
TM$InputFile *.dat
```

A complete list of keywords available with explanation can be found in Turbomatch manual (2007). For easier access one folder on Desktop named as “TM Folder” can be created with two subfolders named e.g. “TM Input” and “TM Output”. Their path should be copied from address bar and pasted into TM$InputFolder and TM$OutputFolder keywords respectively as shown on figure 4.1.

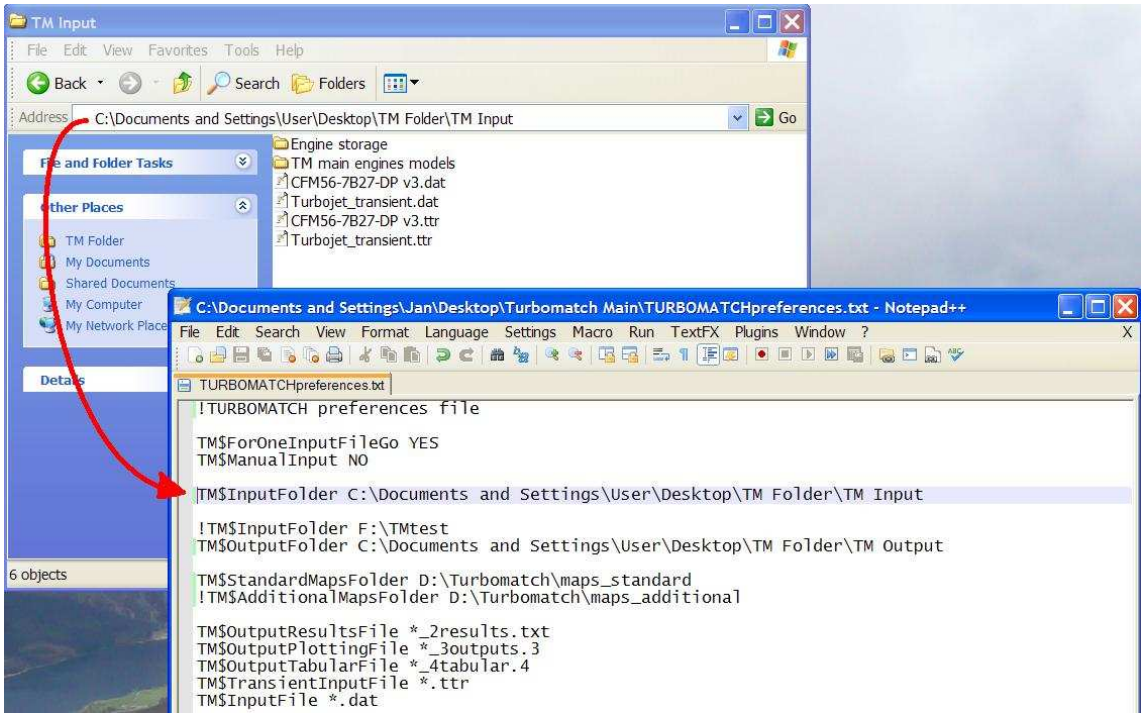


Figure 4.1: User definition of Turbomatch input and output folder location in the preference file

The location of a folder containing component maps should be specified after TM$StandardMapFolder keyword. All engine input files prepared for the simulation need to be placed into “TM Input” folder and after simulation the results will be found in “TM Output” folder.

4.2 Engine Model Input File

Engine model input file for Turbomatch transient version looks almost the same as input file for the legacy version used on Cranfield University network. This subchapter explains the definition of brick data and station vectors for main engine components. Although this version is not sensitive if UPPER or lower case letters are used or if tabs are used instead of spaces it is recommended to use only UPPER CASE letters and spaces. An example of simple turbojet engine model input file can be found in appendix 2.11. The explanation of its sections and description of added features with respect to legacy version is provided:

Selectors defining

```
OD SI KE VA FP
```

1. Case study: DP – Design point simulation only, OD – Off-design study, TR – Transient analysis
2. Units: SI – S. I., IM – Imperial
3. Type of fuel: KE – Kerosene, DI - Diesel HY – Hydrogen, GM – Natural gas (properties according to Manx), GT – Natural gas (properties according to Tomzack)
4. Compressor and turbine geometry: CT – Fixed, VA – Variable
5. Type of print: FP – Full, SP – Short, NP – Minimal, XP – Extra. Generally FP is chosen. Selector XP is used if compressor maps needed in output file

Map definition

```
-1
-1
```

After the selectors the maps of compressor and turbine may be input. Firstly five compressor maps are specified, terminated with "-1", followed by the specification of six turbine maps likewise terminated with second "-1". The prescribed format of component maps can be found in the manual (2007), however for the sake of input file lucidity the compressor maps in the map folder should be changed.

Engine model specification

```
INTAKE  S1,2       D1-6            R100
COMPRE  S2,3       D7-18           R102     V7    V8
BURNER  S3,4       D19-26          R104
TURBIN  S4,5       D27-41                   V28
NOZCON  S5,6,1     D42,43          R110
PERFOR  S1,0,0     D44-47,110,100,104,0,0,0,0,0
CODEND
```

Every keyword represents symptomatic engine component (or brick). Numbers following letter "S" define engine station numbers. If one station number is used in

two different bricks they are connected to each other. Station number “1” should always be used when referring to ambient conditions. Numbers and a defined range of numbers following the letter “D” specify references to brick data storage. Each brick has preselected number of brick data which need to be specified. The number following “R” represents the engine result and the number after “V” or “W” specifies which brick data or station vector is selected as a variable. This section must be terminated with “CODEND” keyword and by four slashes “////” below it.

INTAKE brick data

```
1 0.0            ! INTAKE: Altitude [m]
2 0.0               ! Deviation from ISA temperature [K]
3 0.0               ! Mach number
4 -1                ! Pressure recovery, according to USAF
5 0.                ! Deviation from ISA pressure [atm]
6 0.                ! Relative humidity [%]
```

Two brick data were added:

- Atmospheric deviation from ISA pressure [atm]
- Relative humidity of air [%]

COMPRE brick data

```
7 0.75           ! COMPRESSOR I: Z = (R-R[choke])/(R[surge]-R[choke
8 1.                ! Relative rotational speed PCN
9 8.8               ! DP Pressure ratio
10 0.85             ! isentropic efficiency
11 0.               ! Error selection
12 5.               ! Compressor Map Number
13 1.               ! Shaft number
14 1.               ! Scaling factor of Pressure Ratio – Degradation factor
15 1.               ! Scaling factor of Non-D Mass Flow – Degradation factor
16 1.               ! Scaling factor of ETAc
17 0.1              ! Effective component volume [m^3]
18 0.               ! Stator angle (VSV) relative to DP
```

Five brick data were added before Stator angle:

- Shaft number (Compressors and Turbine on same spool share same shaft number)
- Scaling factor of Pressure Ratio – Degradation factor
- Scaling factor of Non-D Mass Flow – Degradation factor
- Scaling factor of ETAc is (Compressor isentropic efficiency) – Degradation factor

- Effective component volume [m^3] (if =”-1” no component volume is assigned for the component) Although this feature is related to Transient analysis only, it must be input even for steady state simulation in which it s value is ignored

BURNER brick data

```
19 0.05          ! COMBUSTOR: Pressure
20 0.99             ! Combustion efficiency
21 -1               ! Fuel flow
22 0.               ! (>0) Water flow [kg s-1 or lb s-1] or (<0) Water to air ratio
23 288.             ! Temperature of water stream [K]
24 0.               ! Phase of water (0=liquid, 1=vapour)
25 1.               ! Scaling factor of ETAb (combustion efficiency
26 -1               ! Effective component volume [m^3]
```

Five brick data were added:

- (>0) Water flow [kg s-1 or lb s-1] or (<0) Water to air ratio
- Temperature of water stream [K]
- Phase of water (0=liquid, 1=vapour)
- Scaling factor of ETAb (combustion efficiency) – Degradation factor
- Effective component volume [m^3] (Variable for the future, currently it is assumed that the volume of last compressor encompasses also the volume of combustor!)

TURBIN brick data

```
27 0.            ! HP TURBINE: Auxiliary or power output [W]
28 0.6              ! Relative non-dimensional massflow W/
29 0.6              ! Relative non-rimensional speed CN (if = -1, value 0.6 is invoked)
30 0.86             ! Design isentropic efficiency
31 -1.              ! Relative non-dimensional speed PCN (= -1 for compressor turbine)
32 1.               ! Shaft Number (for power turbine, the value “0.” is used)
33 5                ! Turbine map umber
34 -1.              ! Power law index "n" (POWER = PCN^n) If = -1: power constant
35 1.               ! Scaling factor of TF (non-D inlet mass flow) – Degradation factor
36 1.               ! Scaling factor of DH (enthalpy change) – Degradation factor
37 1.               ! Scaling factor of ETAt is (Turbine isentropic efficiency
38 200.             ! Rotor rotational speed [RPS]
39 60.              ! Rotor moment of inertia [kg.m^2]
40 0.2              ! Effective component volume [m^3]
41 0.               ! NGV angle, relative to D.P.
```

Six brick data were added before NGV angle. Compressor number (6th brick datum) has been replaced by shaft number that must be the same as shaft number of compressors on the same spool. Added brick data are:

- Scaling factor of TF (non-D inlet mass flow) – Degradation factor
- Scaling factor of DH (enthalpy change) – Degradation factor
- Scaling factor of ETAc is (Turbine isentropic efficiency) – Degradation factor
- Rotor rotational speed [RPS]
- Rotor moment of inertia [kg.m^2]
- Effective component volume [m^3] (if =”-1” no component volume is assigned for the component)

Last three data are only related to transient analysis. For Steady state case studies they are ignored

NOZCON brick data

```
42 -1.           ! CONVERGENT NOZZLE: Swich set (= "1" if exit area "floats"
                    !                            = "-1" if exit area is fixed)
43 1.               ! Scaling factor
```

Only scaling factor brick datum has been added.

DUCTER brick data

This component is not included in the engine model showed in appendix 2.11 but there was one more brick data added – the effective component volume. However this feature is not yet enabled in Turbomatch, therefore the value “-1” should be used.

Station vector specification

```
-1
1 2 100.0           ! item 2 at station 1 = Mass flow(kg/s)
4 6 1400.0          ! Combustor outlet temperature
-1
```

The first “-1” terminates the input of brick data. Now the input of station vectors, necessary to define engine design point, is expected. Because the fuel flow was not specified in the combustor as a brick data, the combustor outlet temperature has to be

input at this point. Second station vector needed to for engine size definition is the inlet mass flow. Station vector input sequence is terminated with the second “-1”.

After the second “-1” is read by the program the program performs design point calculation. The user may input new values for the brick data and station vectors, separated with “-1”. Every engine model input file should be terminated with “-3” which tells Turbomatch that no other data will be input.

4.3 Transient Input File

The additions to the engine input file mentioned above refer to both steady state and transient simulations. When the program is run in the transient mode, another file for transient simulation settings needs to be created. The file has the same name as engine input file and it differs only by extension, which is generally *.ttr (e.g. if engine input file has name “Turbojet.dat”, transient input file for this engine will have name “Turbojet.ttr”. The extension *.ttr can alternatively be changed in Turbomatch preference file.

Particular sections of the transient input file are showed at the beginning of five following subchapters below which the explanation to the section is provided

4.3.1 The Title Section

```
Turbojet transient file
Purpose: To test Turbomatch transient routines
Author: Jan Janikovic
_____________________________________________________
```

Any text typed at the beginning of the file is considered to be a comment for user and it is not processed by Turbomatch. Title section ends as soon as keyword **CODEIN** is found.

4.3.2 CODEIN – CODEND Input Sequence

```
CODEIN
PRECED       2.     ! No. of preceding DP and OD simulation (initial DP calc.including)
INITIM       0.     ! Initial time [sec]
TRANGE      40.     ! Transient performance simulation duration [sec]
STEPLN       0.001  ! Length of one transient step [sec]
FSCHED       2      ! Type of the fuel
FSTBLE      14      ! No. of records in fuel schedule table
PRINTS      2,50,20 ! Results plotting preferences
BDTRAN      D1,3    ! Brick data which are changed during TR simulation (e.q. D2,27)
SVTRAN              ! Station vectors which are changed during TR simulation (e.q. S5V6)
CODEND
```

Keywords **CODEIN** and **CODEND** delimit the section of file where following transient parameters are defined:

PRECED determines the number of steady state simulations done before transient simulation is started, including the initial design point simulation.

INITIM is the initial time of transient analysis [sec]. This parameter is for output purposes only. The transient simulation time begins with this value.

TRANGE defines the duration of transient performance simulation. [sec]. TRANGE > INITIM

STEPLN defines the time step in transient simulation [sec]. If transient simulation is unsuccessful and off-design converges normally, the time increment (STEPLN value) is too big. For transient simulations with component volumes turned on this value will most likely range between <0.005, 0.00001>. For transient simulations with volumes turned off (by using value “-1” in engine model input file) this value will normally range between <0.1, 0.001>. The smaller the STEPLN value the longer the computer computation time. In some cases when the engine is very complex and all volumes are switched off, longer time step may perform better than a shorter one

FSCHED specifies the type of fuel schedule. If it is set to “1”, the fuel flow is input with respect to the simulation time in the table delimited with DATAIN – DATAEND keywords. 2 –Fuel flow calculated from referred fuel flow against rotational speed (from table provided later)

FSTBLE specifies the number of rows in the fuel schedule table if the table is used. If not, this can have arbitrary value, or it can be omitted.

PRINTS specifies which transient steps will be printed in the result file: 1st number - 1 = Printed time points are specified by following two numbers, 2 = Printed time points are specified in the table delimited with commands PLOTIN and PLOTEND ;2nd number - Number of first consecutive transient time steps printed; 3rd number - Every next print takes place after this number of time steps.

BDTRAN specifies the brick data that are changed during transient simulation. To select brick data 1,2,3,25, following should be written after the keyword: D1-3,25 or D1,2,3,25

SVTRAN defines the station vectors that are changed during transient simulation and can be written for example in following way: S3V6,S5V3 or S3V6S5V3 or S3,V6,S5,V3

4.3.3 Fuel Schedule Table

```
FUELSCHEDULEIN
FS 1
178.5790446   0.015123922
177.2460305   0.01493795
175.9160438   0.014852408
174.1354313   0.014769008
172.2297603   0.014678771
169.7183848   0.014520603
166.5162441   0.014358441
162.2449417   0.014249729
156.6426252   0.014171854
149.8918971   0.014109593
143.5405135   0.013913429
135.8869696   0.013787447
130.4960329   0.013508419
127.550637 0.013119175

0.00       ! Time in sec. to initiate this fuel shedule
135.887    ! Final rotational speed (Nfinal) [rps]
0.22983    ! Final fuel flow (Wff_final)
1150.      ! Limiting COT [K]
FS 2
178.5790446   0.028087283
177.2460305   0.027741908
175.9160438   0.027583043
174.1354313   0.027428158
172.2297603   0.027260574
169.7183848   0.026966835
166.5162441   0.026665676
162.2449417   0.026463782
156.6426252   0.026319157
149.8918971   0.026203529
143.5405135   0.025839226
135.8869696   0.025605258
130.4960329   0.025087064
127.550637 0.024364182

20.00      ! Time in sec. to initiate this fuel shedule
194.854    ! Final rotational speed (Nfinal) [rps]
0.67535    ! Final fuel flow (Wff_final)
1500.      ! Maximium permissible COT [K]
FUELSCHEDULEND
```

At least one fuel schedule table should be input if FSCHED is set to “2”. More fuel schedule tables allow simulation of subsequent acceleration and deceleration (or slower acceleration) as many times as required. The fuel schedule tables must be contained within the keywords FUELSCHEDULEIN and FUELSCHEDULEND. Every new fuel schedule begins with “FS #” where # is the number of the fuel schedule from the top. This is followed by a table, where the first column represents the referred rotational speed and the second the referred fuel flow.

$$\text{Ref. rotational speed} = \frac{N_{shaft}}{\sqrt{\frac{T_{t\,HPCin}}{T_{ISA\,SLS}}}}$$

$$\text{Ref. fuel flow} = \frac{W_{ff}}{\frac{P_{t\,BURNERin}}{P_{ISA\,SLS}}\sqrt{\frac{T_{t\,BURNERin}}{T_{ISA\,SLS}}}}$$

where

$N_{shaft} = PCN \cdot N_{DP}$ [RPS]

$T_{t\,HPCin}$ - Inlet total temperature to high pressure compressor [K]

$P_{ISA\,SLS}$ = 101 325 Pa

$T_{ISA\,SLS}$ = 288.15 K

$P_{t\,BURNERin}$ - Total pressure at the inlet of the combustor [Pa]

$T_{t\,BURNERin}$ - Total temperature at the inlet of the combustor [K]

W_{ff} - Fuel flow [kg/s]

Fuel schedule table is terminated with a blank line followed by four numbers:

1st value below table – Fuel schedule of corresponding table will be applied when transient engine runtime reaches this value

2nd value below table – Final rotational speed – when this speed is achieved, fuel flow will be determined by 3rd value below the table

4th value below table – Limiting combustor outlet temperature (Maximum COT for acceleration, Minimum COT for deceleration)

4.3.4 Table of Times To Plot

```
PLOTIN
0.
0.5
1.
1.5
2.
2.5
3.
3.5
PLOTEND
```

This section is only added to transient input file if PRINTS first value equals “2”. The table determines which time steps should be printed. This option is very helpful when the whole flight is simulated in transient performance. For instance, for parts of the flight where the ambient and/or engine parameters change often denser plots are needed then for those parts where the aircraft is in it’s steady flight.

4.3.5 Variation of Brick Data and Station Vectors during Transient Performance

```
DATAIN
TIME D1 D3
0.   0. 0.
1.   0. 0.1
2.   100. 0.2
DATAEND
```

This is the last section of transient input file which must be always included. It determines how brick data and station vector change according to transient time. All brick data and station vectors that are allowed to change during off-design simulation may be also manipulated during transient. The user has thus the possibility to simulate even cases with varying Mach number during take-off, temporary ambient temperature increase during missile launch, etc.

4.4 A Step by Step Guide How to Produce Transient Curves

Step 1: Engine model preparation

First the engine design point should be created according to the chapter 4.2. First case study selector “OD” should be replaced with “TR”

Let’s assume that the engine design point is the top of climb. If the simulation of take-off performance is required the engine initial state must start from ground idle. This is achieved by several OD cases, specified after initial design point:

```
1 0.                ! Altitude
2 15.               ! Deviation from ISA temperature
3 0.                ! Mach number
107 1.2285
-1
-1
107 1.1141
-1
-1
107 1.0318
-1
-1
107 0.93634
-1
-1
107 0.84622
-1
-1
107 0.76068
-1
-1
107 0.67536
-1
-1
107 0.58727
-1
-1
107 0.50133
-1
-1
107 0.41813
-1
-1
107 0.3449
-1
-1
107 0.27672
-1
-1
107 0.22982
-1
-3
```

Step 2: Transient input file preparation – CODEIN, CODEND

Between two keywords CODEIN and CODEND transient simulation settings are specified. For the following settings the transient simulation of the engine model from 0 to 40 seconds will start after 14 steady state simulations (including the design point analysis). Fuel schedule will be employed and apart from fuel flow the user is also allowed to change the altitude (brick datum 1) and the Mach number (brick datum 2)

```
CODEIN
PRECED       14.     ! No. of preceding DP and OD simulation
INITIM       0.      ! Initial time [sec]
TRANGE      40.      ! Transient performance simulation duration [sec]
STEPLN       0.001   ! Length of one transient step [sec]
FSCHED       2       ! Type of the fuel schedule 1 - Fuel schedule with time
FSTBLE      14       ! No. of records in fuel schedule table
PRINTS      1,50,20
BDTRAN      D1,3     ! Brick data which are being changed during transient simulation
SVTRAN               ! Station vectors which are being changed during transient
CODEND
```

Step 3: Fuel schedule table preparation

In order to plot fuel schedule table a set of off-design simulations has to be run prior transients, ranging from highest to lowest throttle settings of the engine. If 14 values for table are needed then 14 off-design cases must be run with their data plotted in convenient way (e.q. excel sheet). Next, the referred fuel flow vs. referred rotational speed table needs to be calculated for steady state (table 4.1)

Wff	P15	T15	PCN	T8	RefN	RefWff
1.2285	3.27E+06	896.24	1.104	440.51	178.579	0.021605602
1.1141	3.04E+06	873.25	1.0855	432.3	177.246	0.021339929
1.0318	2.86E+06	855.17	1.0693	425.86	175.916	0.021217725
0.93634	2.64E+06	833.28	1.0485	417.87	174.1354	0.021098583
0.84622	2.44E+06	810.86	1.0267	409.59	172.2298	0.020969672
0.76068	2.25E+06	788.05	1.0018	401.59	169.7184	0.020743719
0.67535	2.05E+06	764.52	0.97427	394.57	166.5162	0.020512058
0.58727	1.83E+06	738.51	0.93974	386.68	162.2449	0.020356756
0.50133	1.60E+06	710.91	0.89788	378.7	156.6426	0.020245506
0.41807	1.37E+06	681.55	0.84974	370.42	149.8919	0.020156561
0.3449	1.17E+06	654.07	0.80421	361.8	143.5405	0.019876328
0.27672	9.67E+05	624.53	0.74925	350.41	135.887	0.019696352
0.22983	8.35E+05	601.78	0.71114	342.29	130.496	0.019297742
0.1961	7.45E+05	583.8	0.68815	335.49	127.5506	0.018741678

Table 4.1: Generation of the ref. fuel flow versus ref. rotational speed table.

Transient fuel flow schedule is input as a percentage increase (for acceleration) or decrease (for deceleration) of the referred fuel flow. An increase (or decrease) by 30% may be set initially. If the compressor operating point during transient exceeds surge margin, increase in fuel flow should be reduced (e.q. 25%). The maximum COT (for acceleration) should be set to maximum allowable temperature during Take-off. This value is usually obtained from engine take-off conditions and thrust from public domain. Minimum COT (for deceleration) should be set to lowest possible value where the simulation still can converge. Alternatively it should be set to the value above which flame-out state of the engine cannot occur. The first fuel schedule table “FS 1” will control the fuel flow from the beginning (time = 0). The second fuel schedule table “FS 2” will control fuel flow after 20 seconds of transient simulation:

```
FUELSCHEDULEIN
FS 1
178.5790446   0.028087283      ! Ref. rotational speed N_shaft/sqrt(Tt26/T[ISA SLS]),
Referred fuel flow Wff/(P3/P[ISA SLS]*sqrt(T3/T[ISA SLS]))
177.2460305   0.027741908
175.9160438   0.027583043
174.1354313   0.027428158
172.2297603   0.027260574
169.7183848   0.026966835
166.5162441   0.026665676
162.2449417   0.026463782
156.6426252   0.026319157
149.8918971   0.026203529
143.5405135   0.025839226
135.8869696   0.025605258
130.4960329   0.025087064
127.550637 0.024364182

0.00       ! Time in sec. to initiate this fuel shedule
194.854    ! Final rotational speed (Nfinal) [rps]
0.67535    ! Final fuel flow (Wff_final)
1692.      ! Maximium permissible COT [K]
FS 2
178.5790446   0.015123922      ! Ref. rotational speed N_shaft/sqrt(Tt26/T[ISA SLS]),
Referred fuel flow Wff/(P3/P[ISA SLS]*sqrt(T3/T[ISA SLS]))
177.2460305   0.01493795
175.9160438   0.014852408
174.1354313   0.014769008
172.2297603   0.014678771
169.7183848   0.014520603
166.5162441   0.014358441
162.2449417   0.014249729
156.6426252   0.014171854
149.8918971   0.014109593
143.5405135   0.013913429
135.8869696   0.013787447
130.4960329   0.013508419
127.550637 0.013119175

20.00      ! Time in sec. to initiate this fuel shedule
135.887    ! Final rotational speed (Nfinal) [rps]
0.22983    ! Final fuel flow (Wff_final)
1150.      ! Limiting COT [K]
FUELSCHEDULEND
```

Step 3: Brick data table change

Following code assumes that aircraft reaches Mach 0.1 in 3 seconds and Mach 0.2 and altitude 100m in 10 seconds:

```
DATAIN
TIME D1 D3
0.   0. 0.
3.   0. 0.1
10.   100. 0.2
DATAEND
```

Step 4: The generation of Turbomatch result files for excel

Since the transient study can return several thousands of time plots it would be very laborious to process them manually. The engine results for transient simulation are not plotted into Turbomatch output file *.tmr. They are plotted in tabbed Turbomatch tabular results file *_4tabular.4. This file can be loaded into post-processing utility, such as Tureput, to produce user-friendlier outputs.

Tureput (TUrbomatch REsults Processing UTility) is consol user interface programs, written in Fortran 95, which reads Turbomatch tabular results file and creates a set of excel files. User can then easily create chart of any variables by Excel chart tool. Tureput folder contains two files:

tureput main.exe – Executable file of the program

tureput.opt – Options file of Tureput. This file determines which Turbomatch tabular file will be postprocessed and where excel files will be placed:

```
! Tureput options file

TM_ResultFilePath C:\Documents and Settings\User\Desktop\TM Folder\TM Output
TM_ResultFileName CFM56-7B27-DP v3_4tabular.4   ! Results file name
EngineType 1   ! 1 = Aircraft Engine
               ! 2 = Industrial Engine

Tureput_OutputFilePath C:\Documents and Settings\User\Desktop\TM Folder\TM Output
```

In the above Tureput option file the following information is stored:

Turbomatch tabular results file is placed in “C:\Documents and Settings\User\Desktop\TM Folder\TM Output” folder and has the name “CFM56-7B27-

DP v3_4tabular.4". Results processed are related to aircraft engine and excel output files will be placed to "C:\Documents and Settings\User\Desktop\TM Folder\TM Output" folder

To process Turbomatch results files into MS Excel files the user should:

1. Open Tureput.opt and write Turbomatch tabular results file name after TM_ResultFileName
2. Run "tureput main.exe"
3. Excel engine output files will be located in folder specified in Tureput. opt

Step 5: Plot charts

- The compressor transient working line (figure 4.2) can be plotted from SVdata_results.csv which contains results for engine station vectors
- The thrust variation can be found in Performance_results.csv file.

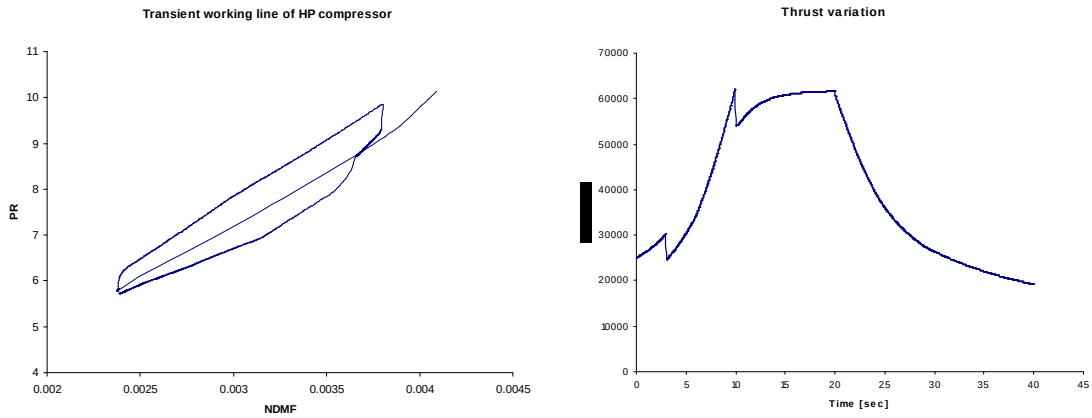


Figure 4.2: An example of HP compressor transient working line and transient thrust variation for a two-spool turbofan for acceleration and deceleration

CHAPTER 5

ENGINE MODELING AND FLIGHT PATH ANALYSIS

5.1 The Motivation for Flight Path Analysis Study

The increase in competitiveness and expansion in aircraft manufacturing industry and in airline companies has called for more advanced cost evaluating methods. Airline expenses can be sub-divided into groups from which the maintenance and partly fuel costs are those expenses that the airline manufacturers can directly influence either by aircraft design or by engine selection. Fuel costs can be calculated relatively accurately by available flight path analyzing software like Hermes or FLOPS (McKinney 1967, Lavelle, Curlett October 1994) using aircraft size and weight data and engine SFC data. According to the analysis the aircraft manufacturer can offer the customer aircraft fitted with engine with better fuel economy. Airline companies then can further eliminate expenses by buying fuel on cheaper airports and if the fuel price distinction between two airports of an air route is too high, the airline can decide to fill the tank with fuel for both onward and return journeys. This, however, results in higher aircraft weight and consequently higher thrust must be used for take-off, climb and cruise, which reduces the engine components life and increases engine maintenance costs and shop visit rate.

Further engine parts life extension can be achieved by applying the derate, that means lower maximum thrust than it is available for the engine, when ambient conditions are and runway length are suitable. Lower ambient air temperature, longer runway length, higher runway friction factor allow the pilot using lower thrust during take-off. To apply the derate is only sensible when it is possible to evaluate its impact on life consumption.

For direct maintenance costs evaluation a lifing model was developed to estimate the life consumption dependencies on used derate, ambient conditions and take-off condition. Prior to life estimation calculation an *engine flight path cycle analysis* is carried out which will provide the variation of thermodynamic parameters during the flight that are important for life estimation (figure 5.1).

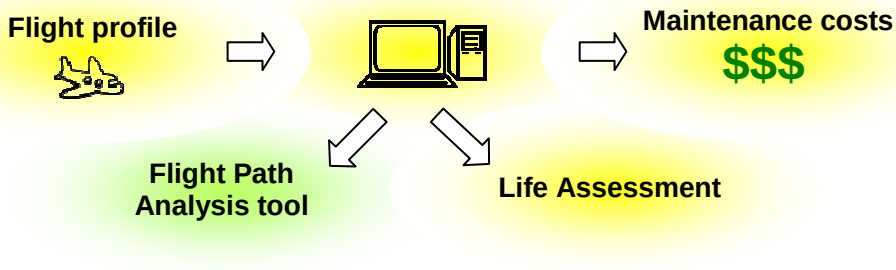


Figure 5.1: The definition of FPA and Life Assessment projects

Engine flight path analysis is divided further into two groups: The engine performance simulation and the flight path analysis. Within the scope of Engine performance simulation an engine model is created and simulated to obtain engine performance data for the flight path analysis. Afterwards an aircraft model is created and its flight cycle simulated across selected flight paths. The choice of selected flight paths determines usually current aircraft air routes or air routes planned for the aircraft in the future.

5.1.1 Aircraft Operating Costs Analysis

Aircraft operating expenses can be divided into several subgroups (figure 5.2 and appendix 5.1), of which the maintenance expenses vary from 6% to 35% of the whole aircraft operating costs (Lawley 2004). From all maintenance expenses the engine maintenance costs represent approximately 50% and the rest appertains to airframe maintenance costs (Maddalon, Nasa 1978). These expenses are of the highest importance for aircraft manufacturers because they indicate the competitiveness of their products. It should not be left out of consideration that fuel expenses also show

the economy of the aircraft and they depend on a range of factors, especially the engine SFC, flight performance and market conditions.

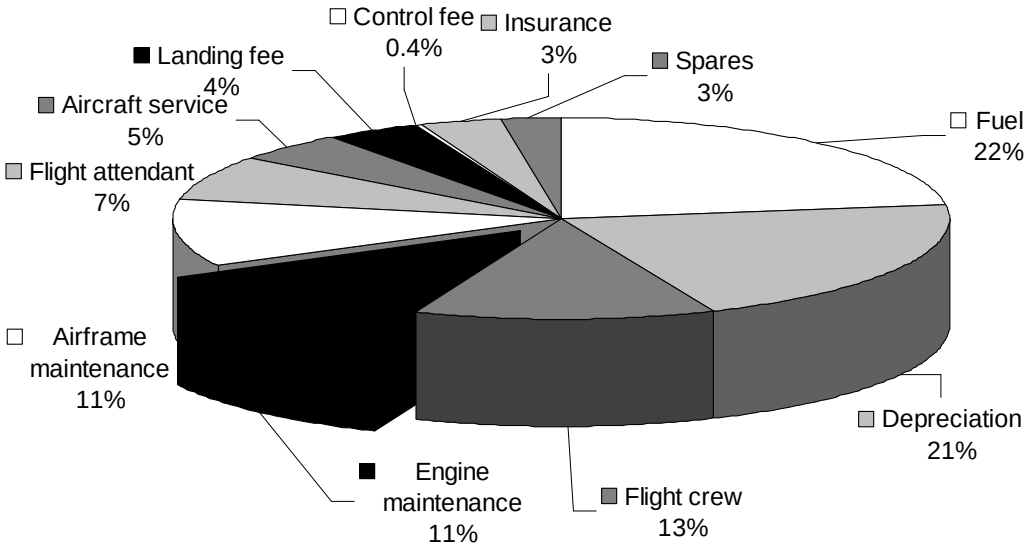


Figure 5.2: Aircraft operating costs breakdown (Maddalon, Nasa 1978)

If a new engine appears on the market the accuracy of analyses done by applying data obtained from different aircraft and different engine would become disputed. Physics based research needs to be undertaken in order to asses the dependence of engine used on engine maintenance costs. By evaluating the life cycle of engine for selected flight paths, the engine shop visit rate can be obtained and used for Engine DMC evaluation. This helps the aircraft manufacturer to choose engine for the aircraft with better economy for selected engine operation profile.

5.1.2 Flight Path Analysis Project Definition

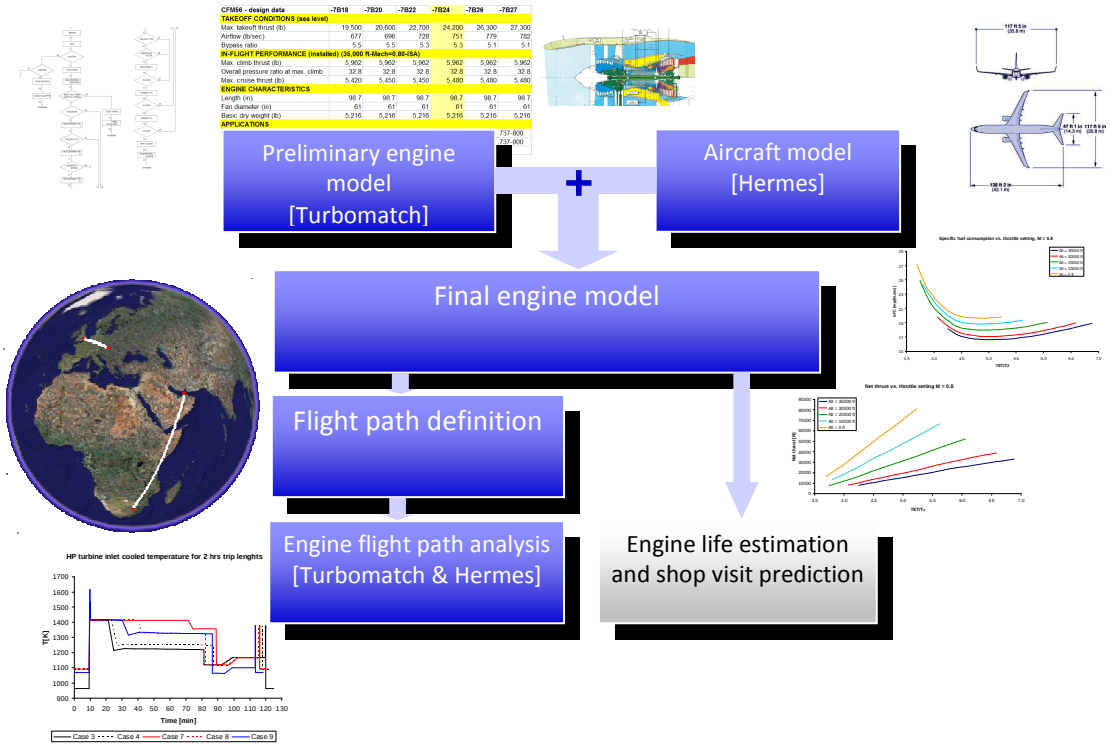


Figure 5.3: The schema of the flight path analysis

In the flight path analysis (FPA) a preliminary engine model and an aircraft model are created. Several reference flight simulations are calculated to ensure that the engine model performance returns expected results. If not, the engine model is adjusted so that the reference flight results are met. The final engine model is thus created. There are two possibilities how the engine life estimation and shop visit prediction are calculated. For the general analysis, the generated engine model and Turbomatch program are directly utilized in engine life estimation and shop visit prediction tool to calculate the severity of selected flight parameters on engine life (Hariharan 2009). For a greater control of studied flight parameters the flight path is first defined externally processed by Turbomatch and Hermes. The resulting thermodynamic parameters are then fed into life estimation and shop visit prediction tool for further analysis. The range of parameters which effect on the engine cycle is usually simulated is shown on figure 5.4.

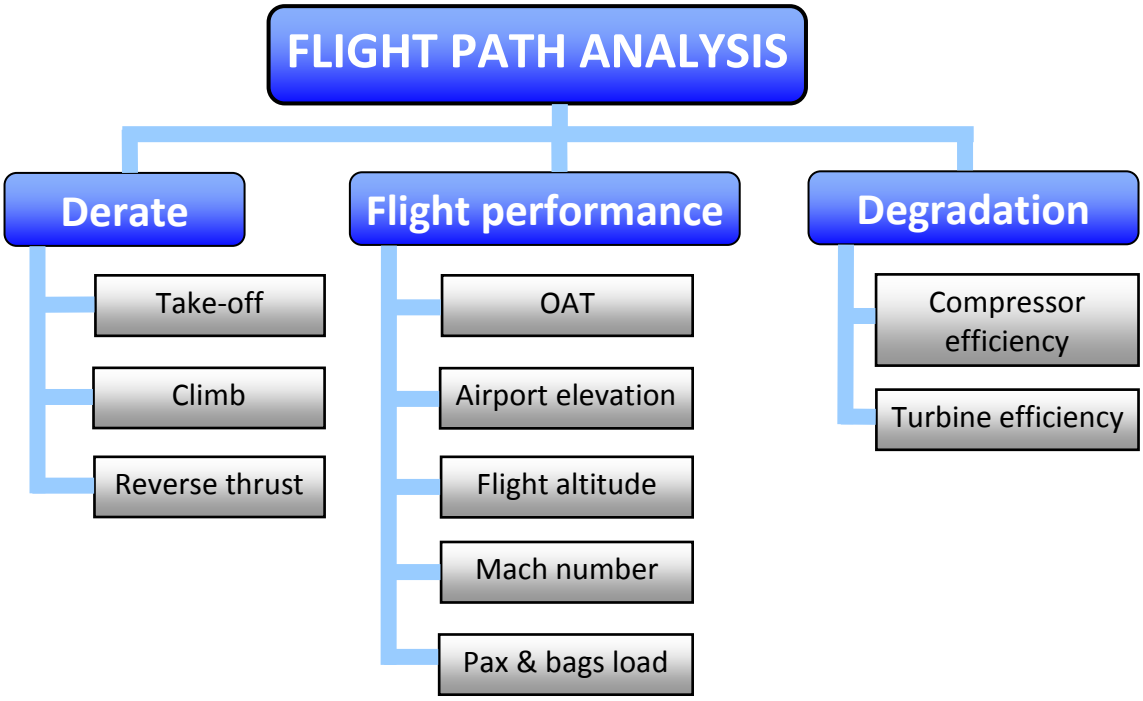


Figure 5.4: The capabilities of flight path analysis

The engine life estimation and shop visit prediction tool processes results from the previous analysis in order to evaluate the life consumption of the engine components throughout individual segments of flight path and define the most critical life-consuming segments of the flight. For instance, for compressor disc life estimation, a technique suggested in (Tong, Halliwell & Ghosn 2004) may be applied. Generally, thermal and structural analysis including weight and size estimations is analyzed first and the life consumption is calculated after for several failure modes like low cycle fatigue, creep and oxidation. It is worth noting that different failure modes are dominant for different flight path segments. In conclusion the life of engine components is estimated using combined damage from the three mentioned failure modes (Hariharan 2009)

5.2 Engine Modeling for FPA

The engine modeling constitutes together with the aircraft modeling the fundamental part of the engine life estimation and shop visit prediction tool. The models provide all the necessary information about the variation of thermodynamic parameters within the engine components. All engine models created were built with a remarkably high degree of complexity in order to capture all necessary air extractions intended for:

- HPT cooling
- LPT cooling
- Shaft cooling
- Operational purposes (cabin heating, pneumatics, etc.)

An engine mass flow diagram (appendix 5.2) is used to create the first draft of engine model component flow chart (appendix 5.3). From the flow chart the engine model input file for Turbomatch is created. It ensures acceptable accuracy for further analysis notwithstanding that input data for the model may not be precisely known at the first stage of analysis. The tool is intended for the aircraft manufacturer, who generally has an access to all the information about the engines which is necessary for the model design. But even when the knowledge of extensive engine data from the manufacturer is limited, a satisfactory model can be created also using information and data from available public domains and literature like Jane's (Gunston, 2004; Jackson, et al. 2006) or FAA or ICAO reports. The design point of the model is created for take-off or cruise rating and then compared to other known operation points. There are several literatures available that deal with engine model creation with limited amount of information (Walsh, Philip 2004; Kurzke 2005)

The program for flight simulation (Hermes) calculates the points of the desired flight path and the engine model is run to produce the variations of thermodynamic and performance parameters throughout the whole flight path.

With respect to the project development several engine model have been designed and studied. They represent the real turbofan engines with two-spool and three-spool architecture used nowadays on commercial aircraft. Their maximum thrust ranges from 121 kN up to 512 kN. Especially one of them has been analyzed in depth, the engine model of a two spool turbofan.

5.2.1 The Engine Model of a Two-Spool Turbofan

The engine model chosen for demonstration of the FPA capabilities is the interpretation of a two-spool turbofan widely used on short and medium range commercial airplanes. Its basic configuration is captured on figure 5.5. HP compressor and the booster (IP compressor) are driven by the HP turbine. The fan is driven by the LP turbine.

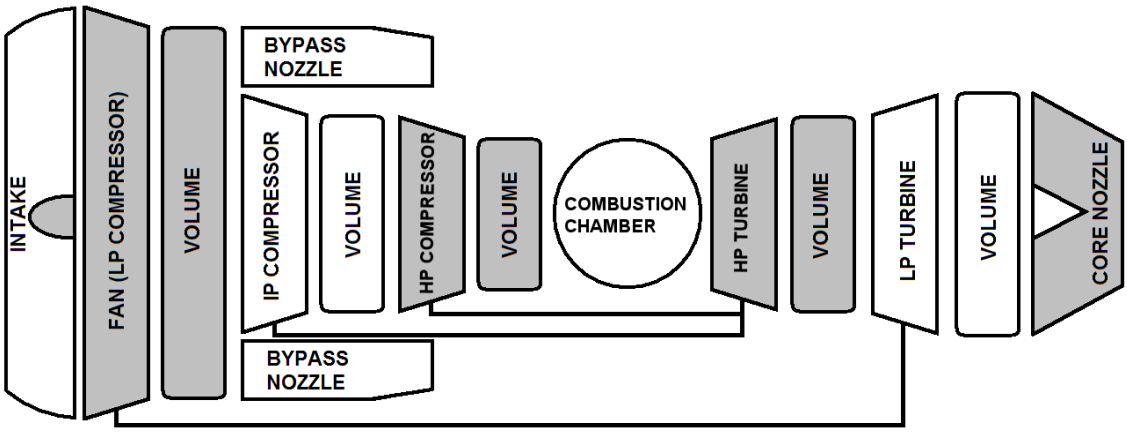


Figure 5.5: The simplified schema of analyzed engine model

Due to several cooling and operational air extractions from the HP compressor, the HP compressor has been split into several segments. Each segment is modeled in Turbomatch with one compressor brick and among them the air extractions are applied. The schema of the final engine model configuration is shown in appendix 5.3. The engine design point is chosen at the top-of-climb ambient conditions with an altitude of 10 000 m and the flight Mach number 0.8.

5.2.2 Engine Performance

After the design of the engine model has been fixed the curves capturing the performance dependencies are plotted. These are excellent to validate the engine model. For example, the right image of figure 5.6 shows the evolution of relative rotational speed with increasing altitude. The curves strongly depend on the component maps. If the predicted trends of the rotational speed does not follow the trends of the real engine (e.g. if the PCN curve for the engine model shows the decrease with altitude, whereas the PCN for the real engine increases), the maps have to be either amended, or replaced by such, that return expected PCN trends.

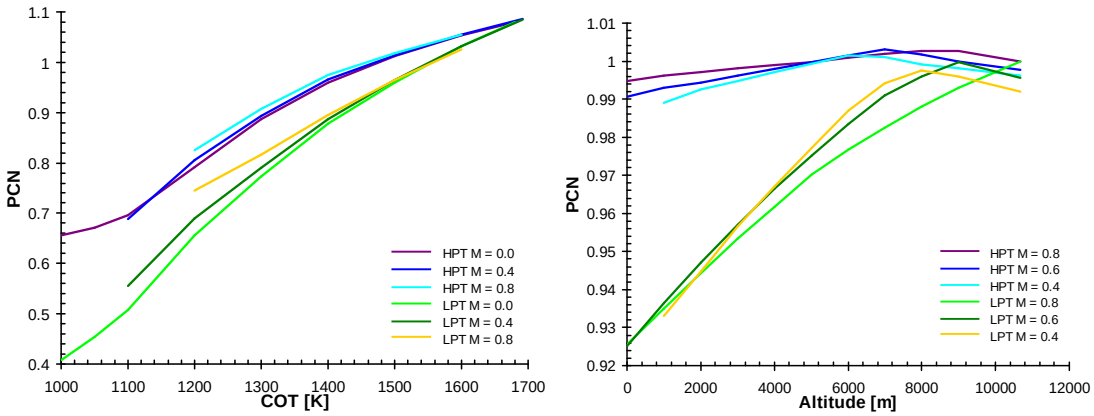


Figure 5.6: The evolution of relative rotational speed. Left: Fixed altitude = 0 m above sea level, Right: Fixed COT = 1440 K (max climb temperature)

Then an extensive engine analysis, capturing possible flight operating points, is carried out. Data of available engine thrust (figure 5.7) and engine specific fuel consumption (figure 5.8) are submitted via a matrix to Hermes, which uses them to calculate the duration of every flight segment and the airplane mass change due to fuel consumption.

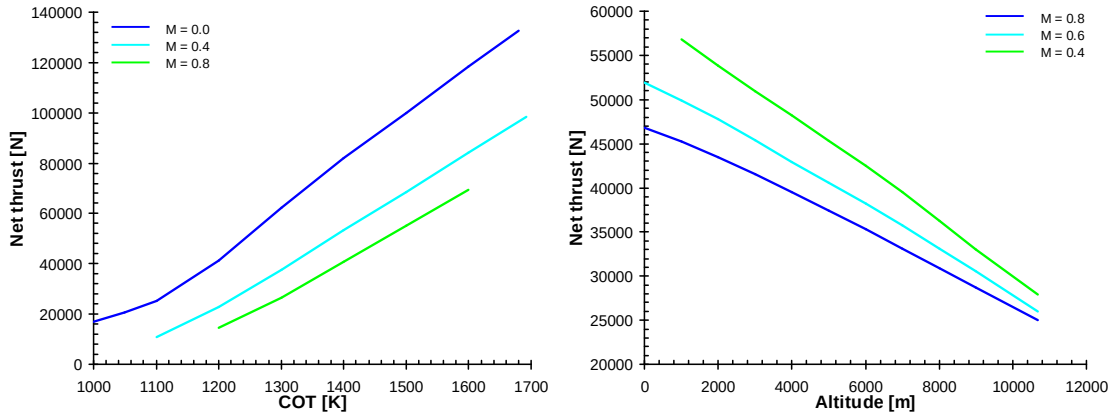


Figure 5.7: The evolution of engine net thrust. Left: Fixed altitude = 0 m above sea level, Right: Fixed COT = 1440 K (max climb temperature)

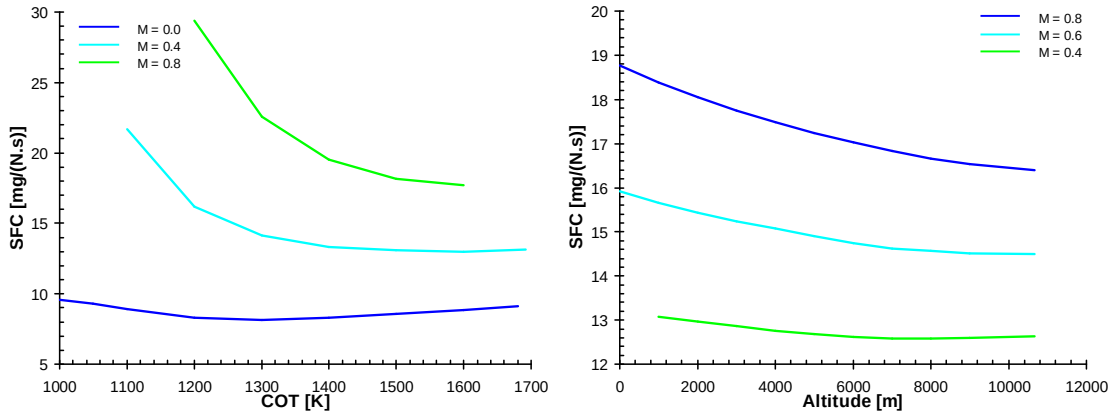


Figure 5.8: The evolution of engine specific fuel consumption. Left: Fixed altitude = 0 m above sea level, Right: Fixed COT = 1440 K (max climb temperature)

When available take-off thrust is calculated the flat rating to maximum thrust is applied. The flat rating is determined was the maximum take-off thrust stated in engine specification documents. In the specification is also mentioned under what ambient conditions can be the maximum take-off thrust achieved. The maximum permissible COT can be calculated for the conditions and for the value of maximum take-off thrust. Then at any engine operating point the lower thrust from the thrust, obtained for the maximum permissible COT and maximum flat-rated thrust determine the maximum engine take-off thrust for relevant conditions (figure 5.9)

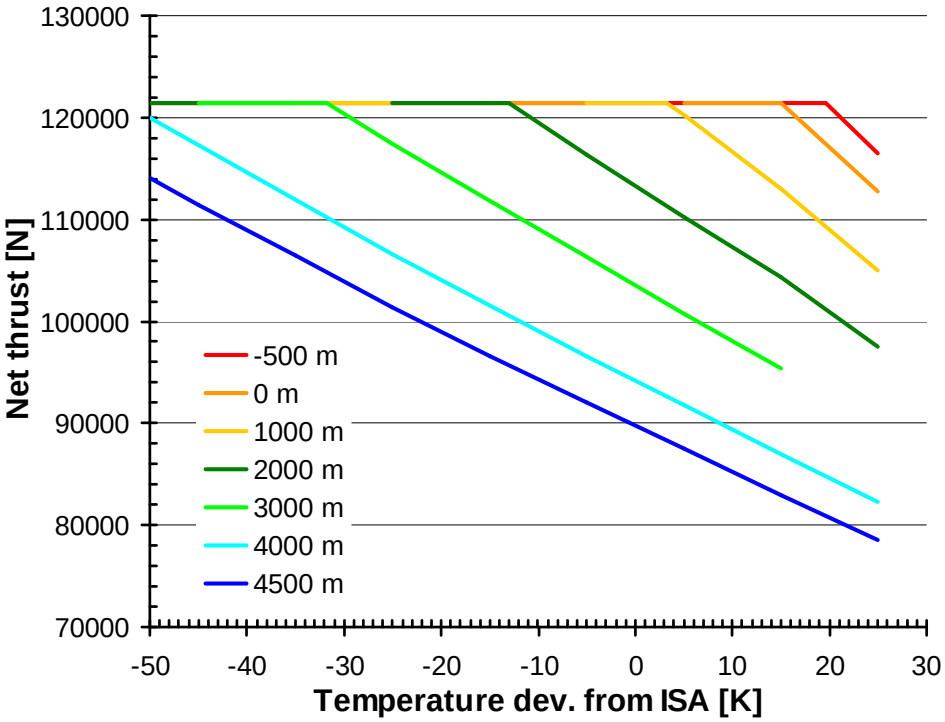


Figure 5.9: The available net thrust for take-off with respect to different runway altitudes

5.3 The Aircraft Model

The synthesis of airplane characteristics is referred to as the aircraft model. In the project the aircraft model has been created according to the accessible aircraft specifications obtained from public domain:

- Wing, tail-plane, fin and fuselage geometry
- Nacelle dimensions
- Landing gear characteristics
- High lift system characteristics
- Masses (MTOW, Payload, fuel, etc.)

The data are processed by Hermes program, that creates the set of aircraft lift-and-drag polars for different aircraft flight configurations. The polars are used in aerodynamic equations (Torenbeek 1982) to calculate the aircraft velocities at take-off

and landing, distances of separate flight segments and the values of the required thrust. Although there are only limited possibilities to replace the engine model, the aircraft model is replaceable if a more accurate aircraft data are available. To accomplish the flight path analysis two more actions are needed – to define the take-off and destination airport and to define the parameters for the flight. For the initial take-off distance calculation the engine and aircraft models are sufficient (figure 5.10)

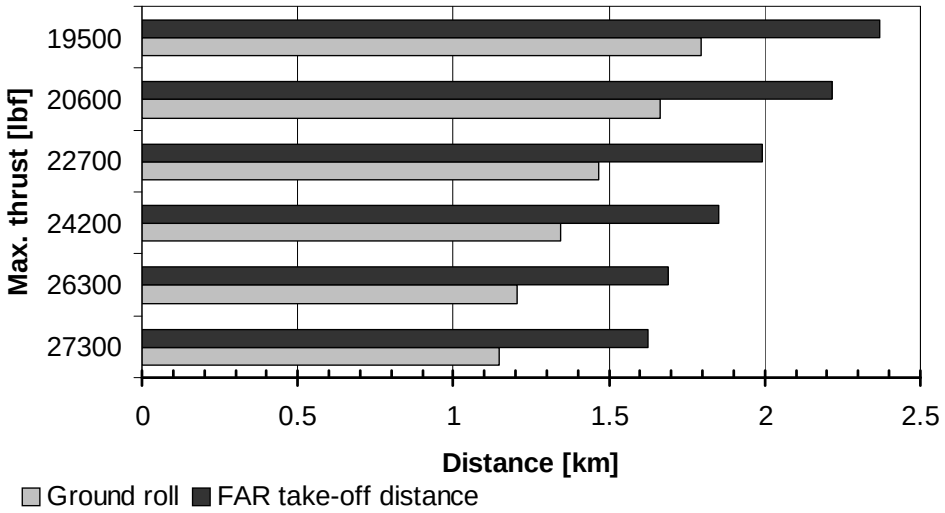


Figure 5.10: The take-off distance for different maximum thrusts at the ambient temperature of 30°C

Another interesting aircraft characteristic that can be acquired from the engine and aircraft model without determining the flight path parameters is the dependence of take-off distance on outside air temperature. Figure 5.11 was generated for a short-haul aircraft at its maximum take-off weight and fitted with two-spool turbofan, discussed in chapter 5.2. The take-off distance increases with the temperature, because the aircraft lift-off velocity increases with the temperature as well and as the aircraft has available the same maximum thrust, longer distance is needed to achieve that velocity. At the OAT of 30 degrees the aircraft power unit reached the maximum combustor outlet temperature. For higher air temperatures than 30°C the maximum thrust reduces, therefore the take-off distance increases at a higher rate.

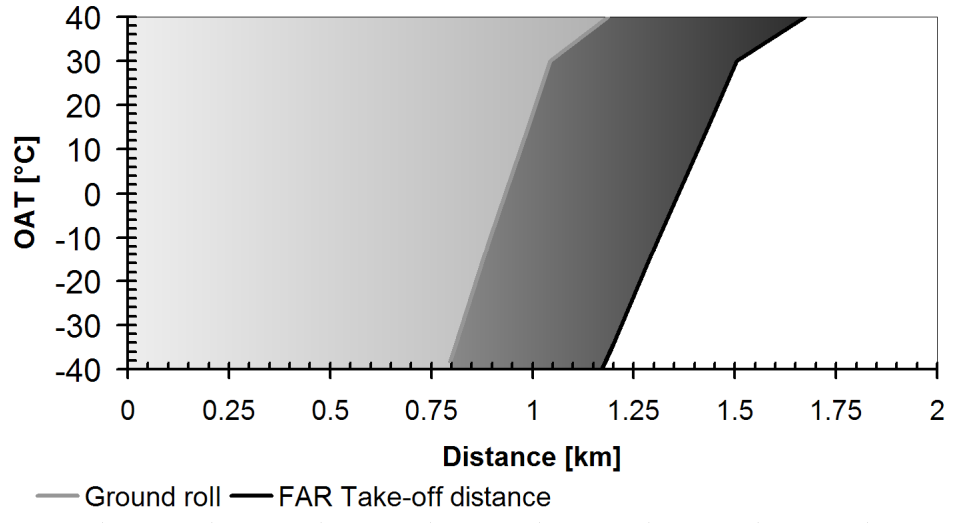


Figure 5.11: Minimum take-off distance with respect to outside air temperature

Payload – range study was also undertaken using the engine and aircraft model. Three studies are shown on figure 5.12. The green and yellow lines were obtained for an aircraft with an increased maximum payload, compared to the BASIC aircraft version, which range capability is illustrated by the red line. The blue line shows the range capability from the aircraft reference guide of the aircraft manufacturer (Anonymous). The simulation curves coincide with reference curve for the settings where maximum TOW of the aircraft is applied. The higher deviation of the simulated and reference curves are most likely caused by different flight profile settings.

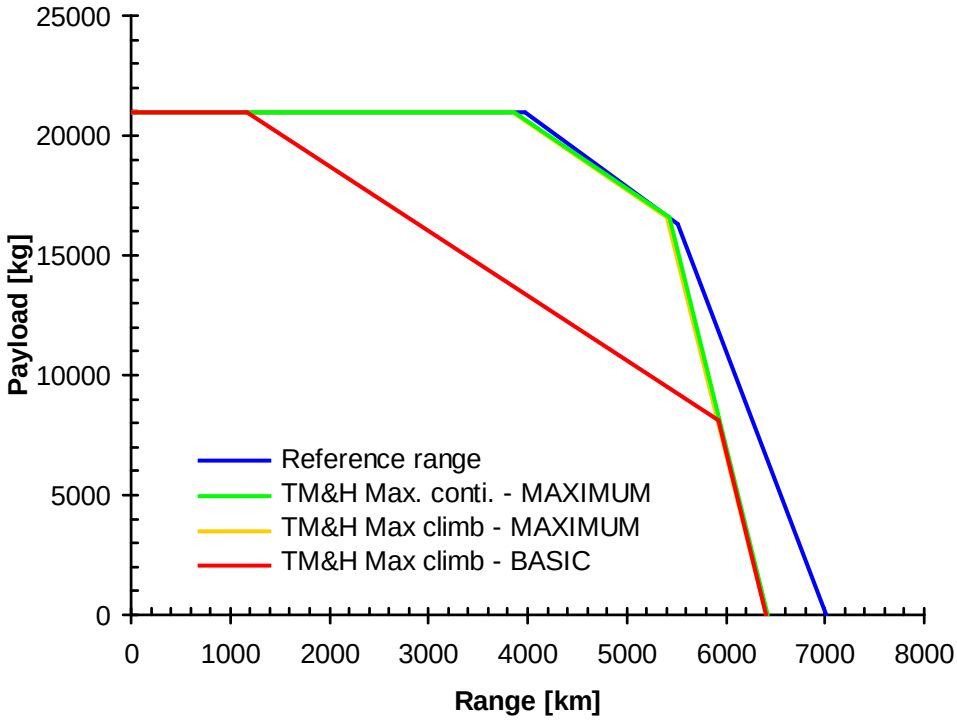


Figure 5.12: Aircraft payload-range capability – comparison between model results and reference values

5.4 Flight Path Definition and Modeling

Under the term "Flight Path Definition" it is not meant that every point of the flight path is defined separately. Flight path points are generated by Hermes according to data about the departure and landing airport and from the parameters of the flight.

5.4.1 Airport Data

- Airport elevation (or airport actual pressure altitude)
- Temperature deviation from ISA
- Runway length
- Friction coefficient

The data about departure and arrival airport are needed to evaluate the thrust requirements during take-off and amount of reverse thrust during landing. Although the calculation of contribution of reverse thrust to aircraft braking power is not included in the model yet, the reverse thrust was assumed to be 90% of the nominal

take off thrust. Airport elevation and runway lengths are found on public domain (Anonymous). Most airports also state the material of runway surface from which the friction coefficient can be determined. The ambient temperature deviation from ISA is calculated from the airport average temperatures (Anonymous). The ambient temperature is particularly important when the effect of derate is studied.

5.4.2 Flight Data

In order to calculate the flight path, following flight parameters need to be specified:

- Distance to be flown
- Flight Mach number
- Temperature deviation from ISA
- Diversion (only if used: altitude, Mach number and distance)
- The initial guess of the fuel weight

Payload is already included in the aircraft model. The initial guess for the fuel weight is adjusted according to the distance flown.

5.4.3 Original Hermes Take-off Modeling

In Hermes (Ogaji, Kyritsis & Laskaridis 2007), the take-off calculation was originally carried out just for single value of aircraft thrust, nevertheless the engine net thrust decreases as the aircraft accelerates. The thrust decrease is predominantly caused by increase in Mach number as aircraft accelerates, resulting in momentum drag increase. This effect is only partly compensated by intake ram pressure increase. A concept of corrected mean T-O thrust was used before T-O model was initially added into Hermes. The foundation of this approach is to calculate a single value of net thrust for which, if it was held constant during ground roll, the time of ground roll during take-off for fixed ground roll distance would be the same as it is for decreasing thrust. If this must be calculated precisely numerical integration should be carried out, however a simplification can be made which assumes linear decrease of aircraft acceleration during T-O caused due to the decrease of net thrust.

$$\frac{da}{dt} = \frac{d^3x}{dt^3} = const. \qquad \text{E5.4.1}$$

$\frac{da}{dt}$ - rate of acceleration change

Then the corrected mean acceleration was evaluated by:

$$a_m = \frac{1}{3}\frac{da}{dt} \cdot t_{GR} + a_0 \qquad \text{E5.4.2}$$

The corrected mean net thrust can be then calculated using following correlation:

$$F_m = \frac{1}{3}\left(F_{LOF} - F_0\right) + F_0 \qquad \text{E5.4.3}$$

Correlation E5.4.3 calculates net thrust which was given a single value to Hermes input file to evaluate the FAR take-off distance. This was only a temporary solution used until Hermes take-off analysis routine has been reprogrammed.

5.4.4 Hermes Take-off and Landing Model update

The idea to upgrade Hermes take-off routine was brought forth almost immediately after this program started to be used for flight path analyzes. Applying the concept of corrected mean net thrust enables more accurate take-off distance calculation (chapter 5.4.3), however for lifing estimation analysis it didn't help at all since the behavior of the engine thermodynamic parameters remained simplified and the whole take-off segment analysis returned only two operating points of the engine – point where aircraft velocity equaled to zero and the point defined as LOF point.

In the first step the whole take-off segment is split into sub-segments defined in CS 25 (Goudou 2003) (appendix 5.4). Length of these segments was already calculated in original version of Hermes where the calculations were based on a single value of net thrust. This procedure remained even in updated routine as a primary guess of take-off distance. In the next step the take-off is divided into 13 take-off nodes:

1. Taxi start
2. Take-off fuel flow taxi

3. Take-off fuel flow increase 1st stage
4. Take-off fuel flow increase 2nd stage
5. Take-off fuel flow increase 3rd stage
6. Take-off fuel flow run
7. Ground roll 1/4
8. Ground roll 2/4
9. Ground roll 3/4
10. Ground roll 4/4 (LOF)
11. Take-off distance 35ft
12. Transition
13. Take-off flight path 1500 ft

Distances, time durations, Mach numbers and aircraft altitudes between two subsequent segments are calculated using basic kinematic equations. These data are then given into Turbomatch engine model to calculate more accurate net thrust pattern. Procedure continues until the overall change in net thrust for all nodes is lower than required.

Similar procedure has been applied to landing as well. On the contrary to take-off, Hermes did not provide any calculations for landing phase. To capture the pattern of thermodynamic quantities of the engine during landing and reverse thrust, the landing breaking section has been split into 11 nodes:

1. Landing start
2. Reverse fuel flow increase start
3. Reverse fuel flow increase 1st stage
4. Reverse fuel flow increase 2nd stage
5. Reverse fuel flow increase up
6. Reverse fuel flow decrease
7. Reverse fuel flow decrease 1st stage

8. Reverse fuel flow decrease 2nd stage
9. Reverse finished
10. Landing end
11. Taxi

This take-off and landing phase segmentation ensure that lifing algorithm will receive more realistic pattern of thermodynamic properties, gradually changing during the take-off and lending phase. Moreover it enables the user to simulate take-off and landing in transient mode of Turbomatch.

5.5 FPA Case Studies

Two example case studies are shown here. Their purpose was to capture the effect of take-off derate on engine performance and thermodynamic parameters. The first study was run in steady state mode, the second in transient. Because the variation of engine parameters during the take-off is significant, the simulation of take-off in steady state mode is rather inaccurate. However the transient mode is sensitive on greater changes in engine and/or ambient parameters. Engine parameters are driven by the fuel control algorithm and therefore they vary smoothly. But the variation of ambient parameters is jumpy (defined from the steady state analysis) therefore it produces rounded steps on transient curves

5.5.1 Take-off Derate – Steady State Analysis

From all analyzed flight segments the highest throttle settings, and thus the highest engines temperatures occur during take-off. Therefore, take-off has the highest impact on the engine hot components' life. Not only the high temperatures, but also the very high change in temperatures cause that the life is consumed considerably. The decrease of the high take-off temperatures has a positive effect on component's life, and one of the options how to do it is to decrease the value of net-thrust during take-off – to apply the derate. Derate is a percentage of the maximum available thrust by which the maximum available thrust has been decreased. In some cases the term derate means the derated maximum available thrust.

Engine manufacturers and airlines are aware that derate has the positive effect on life. If they could evaluate how much the life consumption of the hot components is reduced it is possible to prolong the time interval between shop visits, where the hot components are replaced. Using derates reduces maintenance costs to airline companies so they are the first segment of aircraft industry motivated to use derated thrusts. But the lower thrust available when derate is used also increases the take-off distance. In adverse weather conditions when the runway is wet and the runway friction coefficient is reduced the accelerate stop distance is the first affected. Due to lower take-off thrust the distance between V = 0 and V_{EF} increases and due to the lower friction coefficient the braking distance increases as well. Before the derate is applied, a check must be made if it is safe to do so.

This subchapter shows how engine parameters, that have the highest impact on the engine's life, vary during the take-off. If these parameters are then used in life estimation and shop visit prediction tool, the life consumption estimation may be calculated (Hariharan 2009).

Flight Path definition

Table 5.1 shows the definition of parameters for the analyzed flight. Parameters for both altitudes are identical with take-off and landing ambient temperature of 30°C. The cruise altitude and Mach number are kept constant during the whole flight.

Parameter	Unit	Value
Airport 1		
Elevation	m	0
Runway length	m	2900
Runway surface	-	Concrete
dTISA	℃	+15
TAXI duration	min	10
Airport 2		
Elevation	m	0
Runway length	m	2900
Runway surface	-	Concrete
dTISA	℃	+15
TAXI duration	min	10
Flight definition		
Distance	km	1350
dTISA	℃	+15
Max. altitude	m	10668
CruiseMach number	-	0.78

Table 5.1: The flight definition for take-off derate FPA

The effect of six different derate values is analysed (table 5.2). The take-off distance calculated for the derates is illustrated on figure 5.10.

Derate	0 %	3.7%	11.4%	16.8%	24.5%	28.6%
Net thrust [lbf]	27 300	26 300	24 200	22 700	20 600	19 500
Net thrust [N]	121 436	116 988	107 647	100 975	91 633	86 740

Table 5.2: The set of analyzed take-off derates

In this FPA all parameters except from take-off net thrust has been kept constant. The flight profile calculated by Hermes (figure 5.13) shows the variation of altitude with time.

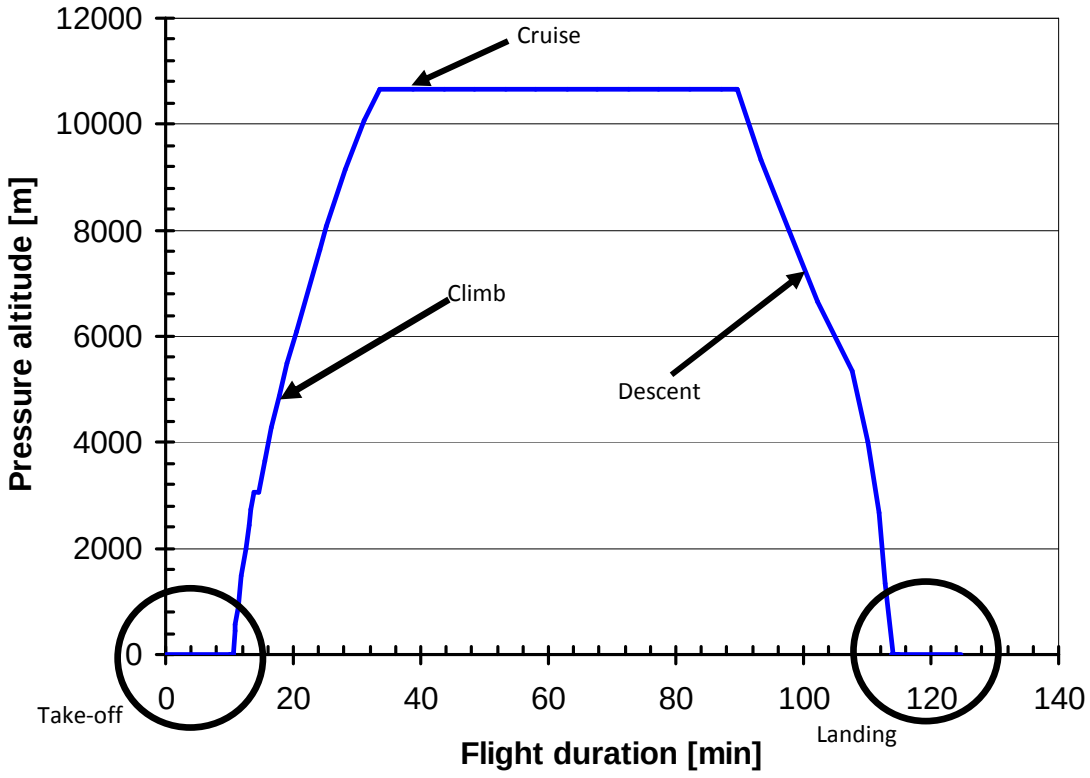


Figure 5.13: The flight profile for the derate FPA

This simulation was run in steady state mode. That means no shaft acceleration inhibition due to rotor inertia took place in the simulation. The main benefit of this approach is the speed of the calculation. The right image of figures 5.14 – 5.17 show the variation of the net thrust and certain thermodynamic parameters of combustor outlet and HP turbine cooling flow through the whole flight against the time scale. Because only derate is analyzed, the parameters are almost equal through the whole flight, apart from take-off and reverse thrust segments. Reverse thrust varies as well since it was set to be 90 % of the take-off value.

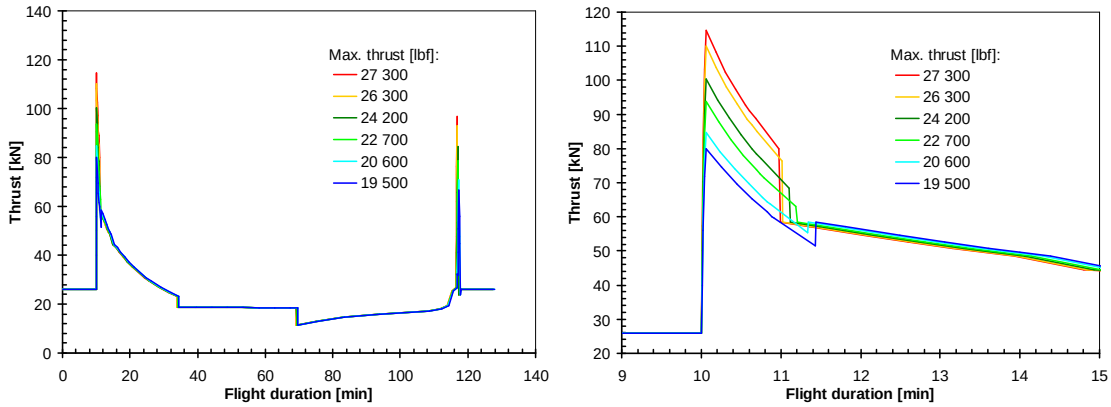


Figure 5.14: The variation of the net thrust through the whole flight (left) and the first five minutes of the flight (right)

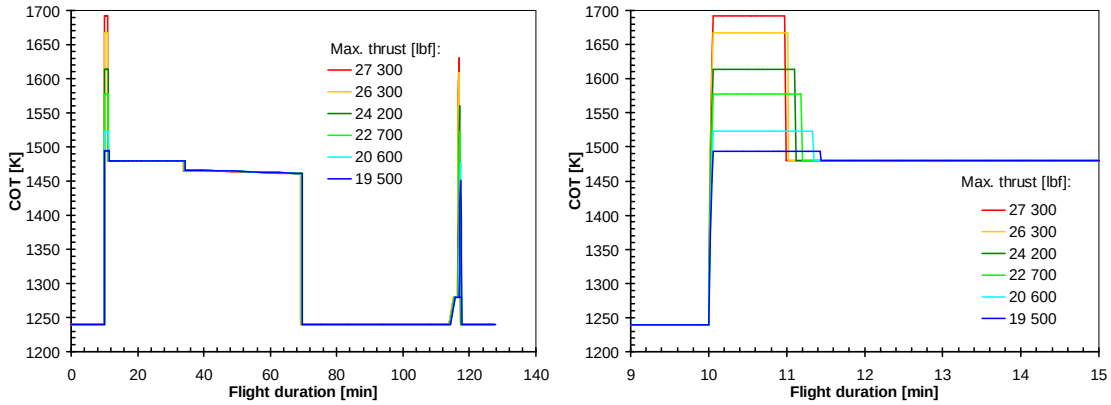


Figure 5.15: The variation of the combustor outlet temperature through the whole flight (left) and the first five minutes of the flight (right)

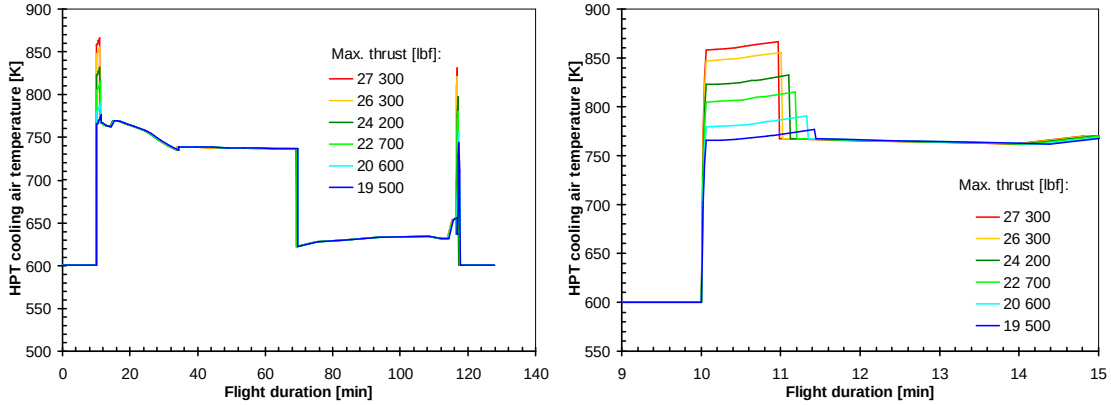


Figure 5.16: The variation of the HP turbine cooling air temperature through the whole flight (left) and the first five minutes of the flight (right)

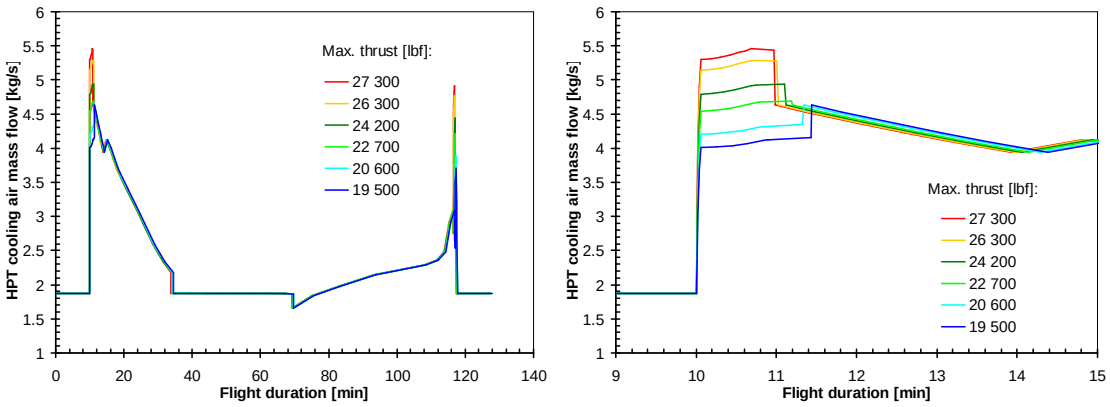


Figure 5.17: The variation of the HP turbine cooling air mass flow through the whole flight (left) and the first five minutes of the flight (right)

5.5.2 Take-off Derate – Transient Analysis

In the previous subchapter the effect of derate on engine parameters was shown. The analysis was calculated in steady state Turbomatch mode. Because the take-off derate has the highest impact on take-off flight segment, it may be sensible to simulate this first flight segment in transient mode, so that the effect of shaft inertia and fuel control system is captured. For the demonstration of the effect in transient, the same flight conditions were used as in chapter 5.5.2. For the sake of clarity, the number of analyzed derates dropped to three (table 5.3).

Derate	0 %	16.8%	28.6%
Net thrust [lbf]	27 300	22 700	19 500
Net thrust [N]	121 436	100 975	86 740

Table 5.3: The set of analyzed take-off derates in transient mode

A five-minute long flight segment has been simulated. The fuel schedule has been fully controlled by the fuel flow control algorithm and the variation of Mach number and pressure altitude was copied from the steady state analysis (chapter 5.3). The variation of COT and Net thrust is similar to steady state analysis. The simulation produce steps wherever a step change in ambient parameters is in input. The change in fuel flow is gradual, because it is calculated dynamically by the fuel control system.

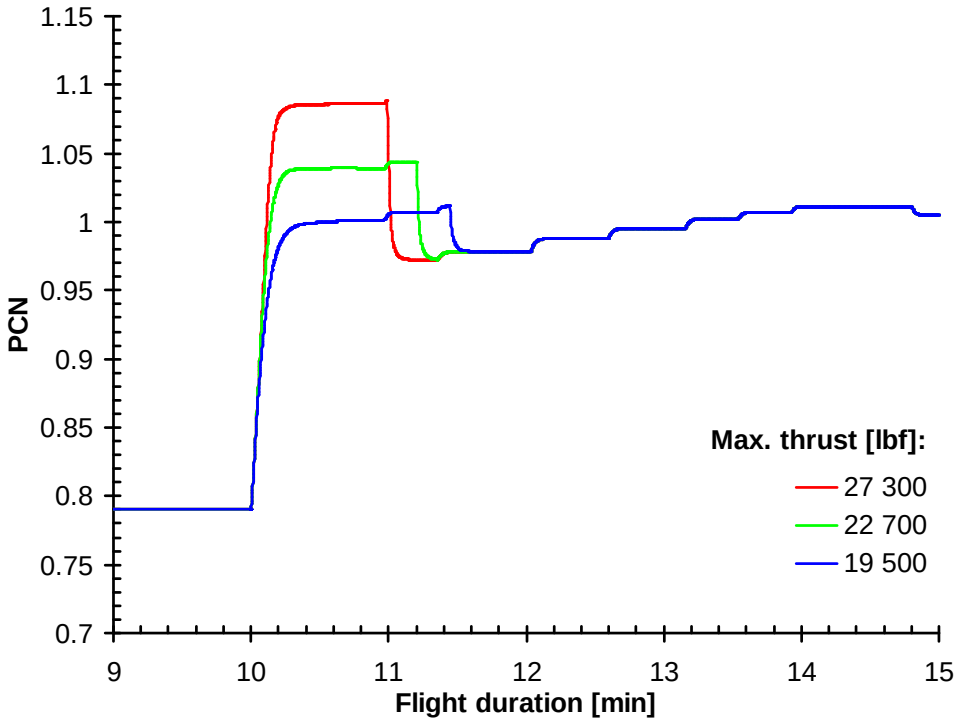


Figure 5.18: The variation of PCN during the take-off and first minutes of climb for the high-pressure spool

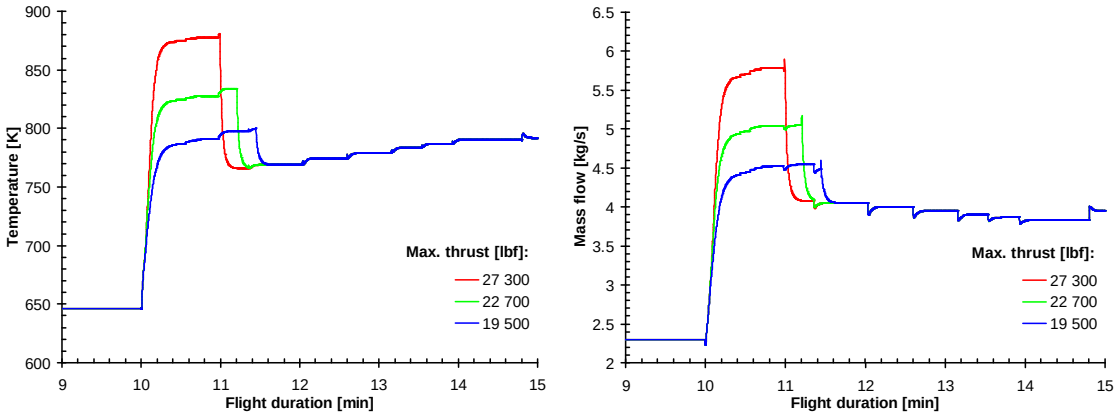


Figure 5.19: The variation of HP turbine cooling air temperature (left) and mass flow (right) during the take-off and first minutes of climb

CHAPTER 6

THE HEAT EXCHANGER MODEL

6.1 Introduction

The efficiency of gas-turbine engines with basic configuration can be improved if an intercooled-recuperated technology (ICR) is used. Intercooler is placed between LP and HP compressor to reduce the inlet air temperature of the HP compressor. Recuperator uses the exhaust heat to increase the inlet air temperature of the combustion chamber thus reducing the amount of fuel used for heating the compressed air. The effect of intercooler and recuperator on the ideal Brayton cycle is shown on figure 5.1

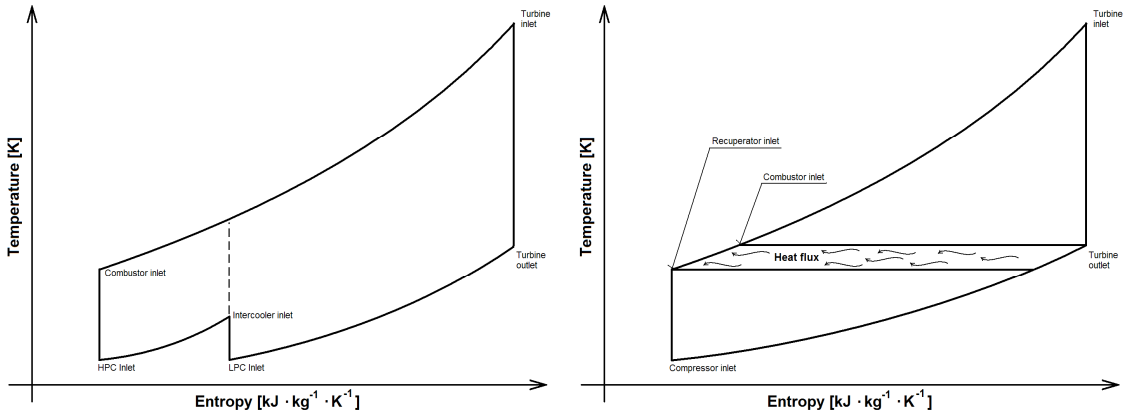


Figure 6.1: The Brayton cycle with intercooler (left) and with a recuperator (right)

6.1.1 The Effect of an Intercooler

An intercooler is generally inserted between two compressors. Its main objective is to increase the specific power for turboshaft engines (Da Cunha Alves et al. 2001), alternatively the specific thrust for future jet engines. The effect of the intercooler on the thermodynamic cycle is shown on figure 5.1. The first, LP compressor, compresses

the inlet air which simultaneously increases the LP compressor outlet air temperature. If an intercooler is implemented, the second compression occurs with colder air which requires less power from the turbine to be extracted from the cycle. This saved power can be used for later expansion in the power turbine, alternatively in propelling nozzle and more power or thrust can be achieved on the expense that more fuel needs to be input into combustion chamber. If an intercooler is used without a recuperator, the thermal efficiency decreases because the heat, of the compression process, that was removed by intercooling is lost and the combustor has to use more fuel flow to maintain the same outlet temperature. Since intercooling reduces the compressor temperature, higher pressure ratios may be used, reducing the loss in efficiency. But the maximum overall pressure ratio should not exceed the optimum value (Da Cunha Alves et al. 2001). If the value of optimum OPR value is exceeded the engine thermal efficiency begins to drop. The positive effect of intercooling on the engine efficiency is the reduction of HP turbine cooling air temperature. The cooling air is extracted after the last stage of HP compressor. The lower the turbine cooling air temperature, the higher the cooling efficiency, so less cooling air can be used.

6.1.2 The Effect of a Recuperator

A Recuperator is a device which uses otherwise wasted heat from an engine exhaust to increase the temperature of compressed air before it enters the combustion chamber. The advantage is the reduction of fuel needed to increase the temperature in the cycle although there is a restriction imposed on the compressor outlet temperature must be lower then turbine exhaust temperature (figure 5.1). This consequently imposes the restriction on maximum value of compressor pressure ratio which determines the increase of compressor outlet temperature (E2.4.8). The heat transfer between the fluids is predominantly convective.

6.1.3 The Applications of Intercooled-Recuperated Technology

Until recently the intercooling-recuperating technology was solely used in industry, where the big size and weight of the devices did not constitute a problem. The

exchange of heat between hotter and colder streams is undertaken in heat exchangers. For land-based applications two types of cooling is used:

- Wet cooling system
- Dry cooling system

In the wet cooling system the cooling medium is water or eventually other liquid. On locations where water source is easily available the intercooling may be performed in cooling towers. Dry cooling is used in applications where water is not available. The cooling medium is the ambient air which flows around lamellas with hot fluid and carries away the “unwanted” heat. The air motion is usually forced by large fans. An example of the industrial intercooled engine is the GE-LMS100 with the thermal efficiency of 46% to 52% if steam injection is used (Reale 2006).

The benefits of intercooled-recuperated cycle on the engine efficiency and power made the technology also interesting for transport application. The traditional heat exchanger design is shell-tube and plate-fin (figure 6.2). For any transport application these heat exchangers are rather large, but using modern manufacturing technologies, such as chemical etching, may reduce their size and weight significantly. Chemical etching is used for the production of heat-exchanger with printed circuit.

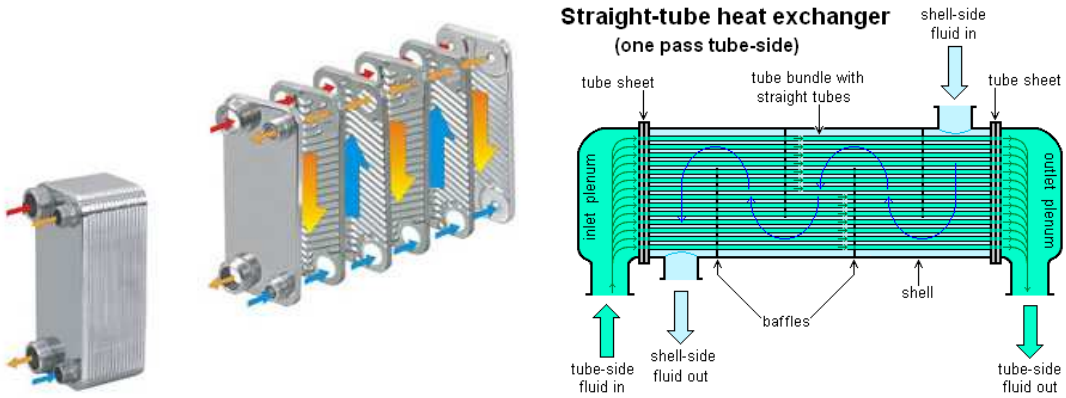


Figure 6.2: The schema of a plate-fin (left) and shell-tube (right) heat exchanger. (Image courtesies: GEA PHE Systems North America, Inc. and H Padleckas)

In the marine industry the intercooler can utilize huge amount of sea water can as the cooling medium. The Royal Navy Air Defence Destroyer Type 45 (figure 6.3) uses Rolls Royce WR-21 (figure 6.4) gas turbine engine to drive the electrical generator which feeds screw-propeller motors.

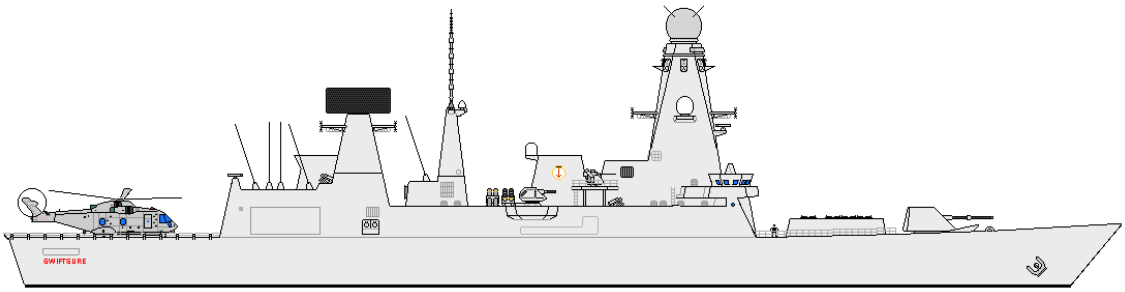


Figure 6.3: The Royal Navy Destroyer Type 45 (Image courtesy: www.defense-aerospace.com)

The WR-21 is an aero-derivative marine gas turbine engine. The inlet HP compressor air is cooled in a printed circuit heat exchanger with high effectiveness, where the cooling medium is a substance of water and ethylene glycol, which has better heat transfer properties than the sea water. The cooling medium is cooled by the sea water. The heat from the exhaust is capitalized in recuperator (Figure 6.4)

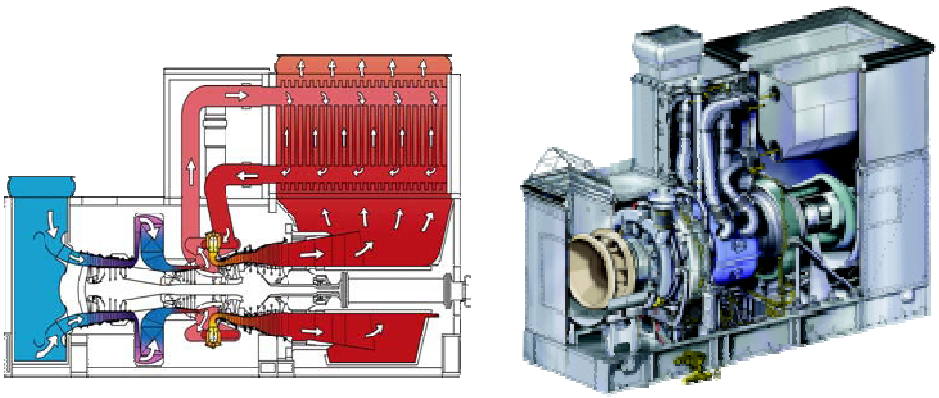


Figure 6.4: WR-21 intercooled-recuperated gas turbine engine (Image courtesy: Rolls Royce)

The WR-21 delivers the maximum power of 25.2 MW. Its mass is 49 800 kg, which makes its mass-to-power ratio 1.98 kg/kW. The S.E.M.T Pielstick PC2.6(B), one of the strongest marine medium speed diesel engine, delivers the maximum power of 6 to 15 MW. The mass-to-power ratio of this engine is from 9.1 to 11.6 kg/kW, approximately five times higher than a ICR gas turbine.

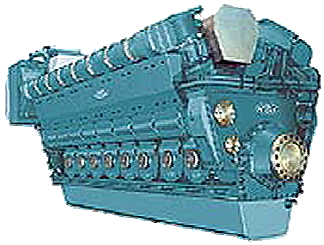


Figure 6.5: S.E.M.T. Pielstick PC2.6(B) marine diesel engine (Image courtesy: Rolls Royce)

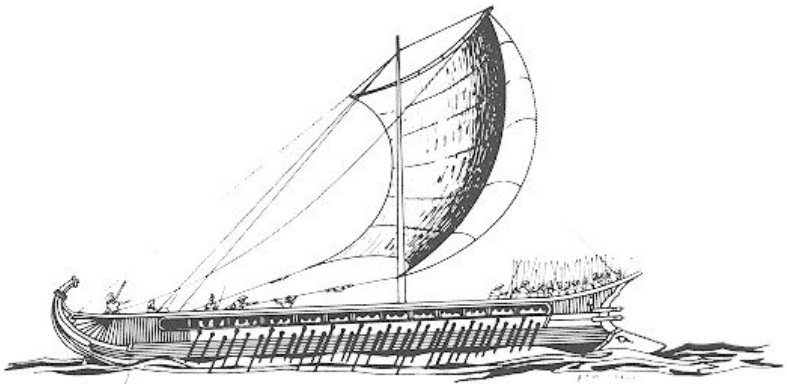


Figure 6.6: Just for the sake of comparison, the average Greek trireme rower was able to deliver approx 350W of maximum power for one hour. Human mass-to-power ratio is thus approx. 200 kg/kW.

The application of intercooled-recuperated technology in aero industry was left out of consideration due to restricted size and weight of heat exchanger components. But as the technology and materials become more sophisticated a research about aero-engines with IRC cycle has been started. A possible design of such gas turbine engine was already presented by MTU (Boggia 2005) and it is shown on figure 6.7. Future heat exchanger may also have for instance helix design and micro-printed circuit.

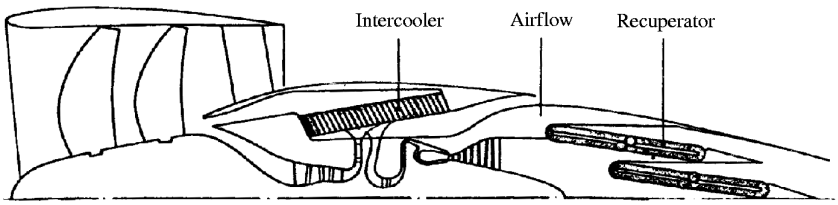


Figure 6.7: A possible configuration of an intercooled-recuperated aero-engine (McDonald et al. 2008)

6.2 ICR Engine Thermodynamic Cycle Modeling

Because heat exchangers have been used in industry for a period of time, the tools to simulate the thermodynamic cycle and performance are available. Turbomatch Legacy was also able to predict the steady state behavior for design point and off-design analysis for engines involving heat exchanger technology. When looking at possible ICR technology application in aircraft industry, a question arises: How the large volumes of heat-exchanger influence the engine dynamic behavior? Industrial engines operate most of the time under steady conditions and the requirements for the transient prediction are not as strict as for the transport gas turbines, especially aero-engines. An aircraft engine undergoes several transients within one flight path cycle. Especially during take-off and landing the knowledge about engine response on parameter change is crucial. In fact, the aircraft engine operates during the whole flight path in transients, but if minor changes are neglected, or they occur over a long period of time, the engine can be simulated in steady state mode.

6.2.1 Basic Consideration of the ICR Transient Performance Method

In order to calculate the transient performance of a gas turbine engine a method for dynamic response of heat exchangers has been developed (Pellicer 2007; Pastor 2007), improved with volume dynamics and heat storage and implemented into Turbomatch TR. The method is general for any ICR engine configuration (chapter 3.1.2). The essence of the method is to “draw” n elements on the heat exchanger cross-section. Each element is assumed to have the same temperature and heat transfer occur only between two adjacent elements (figure 6.8). The advantage of the method is its speed, stability and the capability to simulate the dynamic response of the heat exchanger for

a change of any parameter. During the method development, an attempt to use an adaptation of transient method shown by Abbasov (2006) has been made. The method however had stability problems for the temperature gradients and time steps used. Therefore, a new method has been developed.

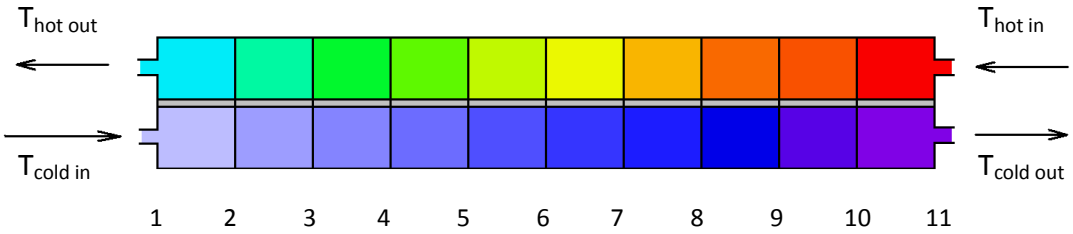


Figure 6.8: The elements of an intercooler

6.2.2 The Classification of Heat Exchanger

Since heat exchangers are used widely used in the industry, there is a great variety offered on the market. Of course, not all of them are used in the gas turbine application, but even when focusing on intercooling and recuperating heat-exchangers the variety of design and technology is great. The most commonly used heat exchangers in terms of construction are:

- shell and tube heat exchangers (figure 6.2 right)
- plate and fin heat exchangers (figure 6.2 left)

Shell and tube heat exchangers have the advantage of being cheaper, therefore they are widely used in the industry. Plate and fin heat exchangers are on the other hand up to 5 times lighter than shell and tube. The fluid flows in the sandwiched passages where a larger surface is in contact with the plate through which the heat is transferred from one stream to the other. The plates are made from aluminum alloys which constitute lighter option compared to steel and have better heat transfer properties. Plate and fin heat exchangers have straight, spiral or helix configuration.

In terms of fluid flow direction the heat exchangers are:

- Parallel flow
- Counter flow
- Cross flow

In counter flow heat exchanger higher temperature difference between inlet and outlet can be achieved (figure 6.9). But if the thermal capacity of one fluid is several times greater than the capacity of the second fluid, the difference is small. This applies for instances where usually a liquid in one passage cools the gas in the second passage down.

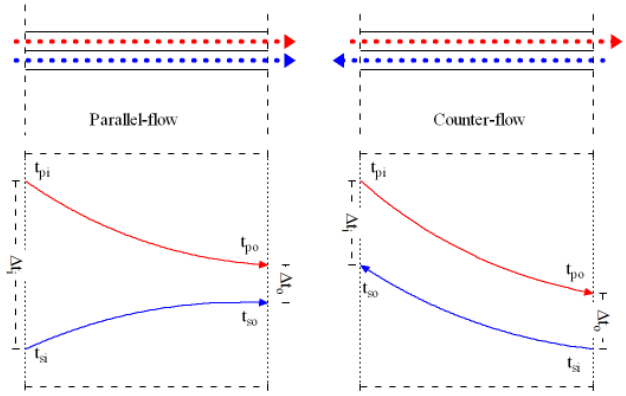


Figure 6.9: The temperature distribution in parallel flow and counter flow heat exchangers (Source: www.engineeringtoolbox.com 2005)

A cross flow heat exchanger is actually a unique kind of counter flow heat exchanger. In figure 6.10 it demonstrated why. It can be observed that the cold fluid flows upwards, whereas the hot fluid flows downwards.

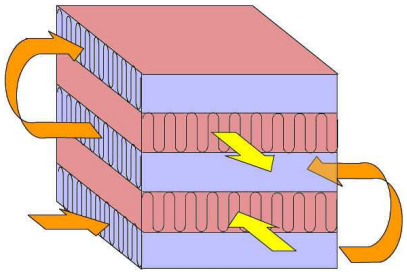


Figure 6.10: Cross flow heat exchanger (Source: http://faculty.kfupm.edu.sa)

6.2.3 Requirements on Transient Algorithm

Because of the diversity in heat exchanger design, created transient algorithm must be universal for any kind of heat exchangers. Several constraints and assumptions are imposed on the developed model:

- The heat transfer between a heat exchanger and surrounding is neglected
- Passages of the flow are simplified to single passage. Heat transfer surface area is distributed uniformly on each fluid side
- The temperature pattern at any fluid cross section is uniform
- Wall thermal resistance is distributed uniformly across the whole heat exchanger
- Fluid does not experience any phase changes
- Longitudinal heat conduction is neglected in both fluids and the wall
- The temperature of both wall surfaces which come to contact with the fluid is the same at every cross section
- Heat transfer coefficients are constant along the heat exchanger
- Constant fluid velocity at the passage cross section and along the heat exchanger
- homogenous flow

The last, but not less important restriction on the algorithm was to keep it robust and stable under any conditions. The algorithm was created separately for counter flow and parallel flow heat exchanger.

6.3 The Heat Exchanger Algorithm for Gas Turbine Thermodynamic Cycle Simulation

The original heat exchanger model in Turbomatch was only able to calculate the steady state behavior. After the Turbomatch has been rebuilt with transient performance simulation capability a requirement of upgrading the heat exchanger model arose. A new heat exchanger model (Pellicer 2007; Pastor 2007) gave the foundations for heat exchanger design, off-design and transient simulations. The algorithm will be shown here, but for theoretical background the references should be used. The volume dynamics and heat storage have been added to the model. Iterative procedures were adopted for parameters not known at the beginning of the simulation. The heat exchanger model starts with the design point after which the sizing of heat exchanger is carried out. Than either design point, off-design or transient simulation may be performed. Theses (Pellicer 2007, Pastor 2007) provide detailed discussion about the model. Further subchapters describe the analytical issues of the algorithm. The new bricks representing the new heat exchanger model were named as HEXCOL (the cold side of the heat exchanger) and HEXHOT (the hot side of the heat exchanger). Original bricks HETCOL and HETHOT were preserved and may be used for steady state simulation. For a recuperated engine model, the HEXHOT brick is placed before the combustor and HEXHOT after the last turbine. When the model carries out the analytical calculations in HEXCOL, parameters of the second hot stream need to be known. But since they are evaluated later in HEXHOT, the hot stream parameters are guessed in HEXCOL and compared with calculated parameters in HEXHOT. The iteration continues until the guessed and calculated values match. For an intercooler the brick COOLER has to be used.

6.3.1 The Design Point Model

The design point model must precede any further calculation. Table 6.1 shows which of the required heat exchanger parameters should be provided by the user. Only the temperature calculation across the heat exchangers is shown here, because the values off mass-flow and pressure ratio are assumed constant along the whole heat

exchanger. But the values do vary in transient mode when the control module routine is applied.

RECUPERATOR	INTERCOOLER
Fluid flow type (counter or parallel flow)	Fluid flow type (counter or parallel flow)
Effectiveness	Effectiveness
Pressure loss in the cold passage	-
Pressure loss in the hot passage	Pressure loss in the hot passage
Design Mach number of the cold flow	-
Design Mach number of the hot flow	Design Mach number of the hot flow
Aspect ratio of the hot passage	Aspect ratio of the hot passage
-	Type of the cooling medium
-	Cooling medium inlet mass flow
-	Cooling medium inlet total pressure
-	Cooling medium inlet total temperature
-	Velocity of the cooling medium flow
Average wall thickness	Average wall thickness

Table 6.1: User-defined parameters

The heat exchanger is simplified to a two-passage model (figure 6.11 and 6.12) where all constraints from 6.2.3 are applied. The passages are divided into n = 10 elements and the boundary between two elements of the same passage constitutes the node in the model. There are 11 nodes used, but there is no restriction imposed on this number and it may be increased if necessary.

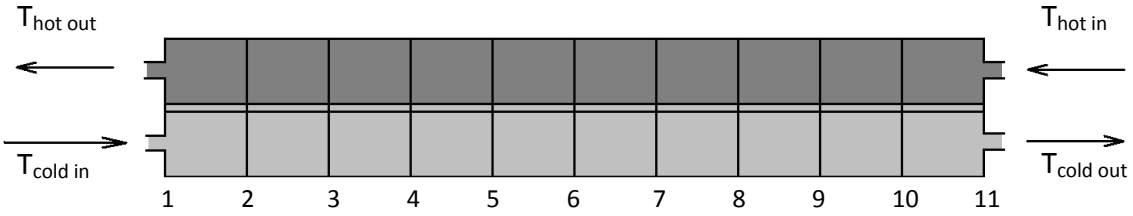


Figure 6.11: The temperature relation and nodes of a cross flow heat exchanger model

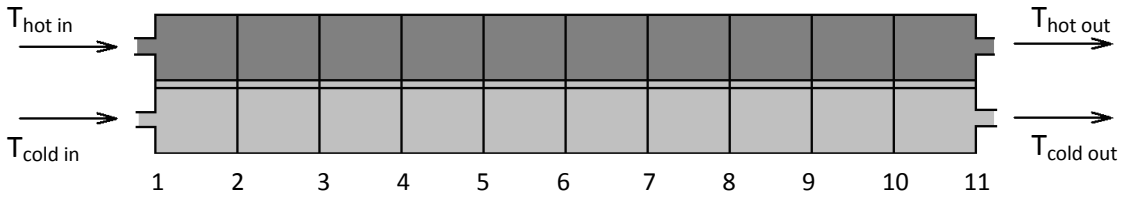


Figure 6.12: The temperature relation and nodes of a parallel flow heat exchanger model

The heat exchanger design parameters are input by the user (table 6.1). The program provides the value of thermodynamic parameters at the cold flow inlet for the recuperator or at the hot flow inlet for the intercooler. The inlet parameters of the cooling medium for the intercooler are known, for the hot flow recuperator inlet the parameters are guessed. The ratio of heats is determined as:

$$CR = \frac{\left(W \cdot c_p\right)_{\min}}{\left(W \cdot c_p\right)_{\max}} \qquad \text{E6.3.1}$$

Where $\left(W \cdot c_p\right)_{\min}$ is the smaller value from $\left(W \cdot c_p\right)_{hot}$ and $\left(W \cdot c_p\right)_{cold}$. The number of transfer units is determined for each type of fluid flow separately:

Counter flow: $$NTU = \frac{1}{CR - 1} \cdot \ln\left(\frac{\varepsilon_{HX} - 1}{\varepsilon_{HX} CR - 1}\right) \qquad \text{E6.3.2}$$

Parallel flow: $$NTU = -\frac{\ln\left(1 - \varepsilon_{HX} \cdot \left(1 + CR\right)\right)}{1 + CR} \qquad \text{E6.3.3}$$

From the number of transfer units the UA factor is evaluated:

$$UA = NTU \cdot \left(Cp \cdot W_{in}\right)_{\min} \qquad \text{E6.3.4}$$

Knowing NTU and UA, the temperature pattern of the whole heat exchanger may be evaluated.

Counter flow heat exchanger

For the counter flow heat exchanger the $T_{cold\ n} = T_{cold\ in}$ and the $T_{hot\ 1} = T_{hot\ in}$ (figure 6.11). The outlet temperature from the cold passage is calculated:

$$T_{cold\ 1} = T_{cold\ out} = \frac{1}{e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right]\cdot UA} - \frac{(W\cdot c_p)_{cold}}{(W\cdot c_p)_{hot}}} \cdot$$
$$\cdot \left\{ T_{cold\ in}\left[1 - \frac{(W\cdot c_p)_{cold}}{(W\cdot c_p)_{hot}}\right] + T_{hot\ in}\left[e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right]\cdot UA} - 1\right]\right\} \qquad \text{E6.3.5}$$

Then the temperatures of the hot and the cold flow are calculated for every node. Index i refers to the node number.

$\forall T_{hot\ i},\ i \in \left]2, n\right[:$

$$T_{hot\ i} = \frac{1}{1 - \frac{(W\cdot c_p)_{hot}}{(W\cdot c_p)_{cold}}} \cdot \left\{ \begin{array}{l} T_{cold\ in}\left[e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right]\cdot UA\frac{(i-1)}{n}} - \frac{(W\cdot c_p)_{hot}}{(W\cdot c_p)_{cold}}\right] \\ + T_{hot\ in}\left[1 - e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right]\cdot UA\frac{(i-1)}{n}}\right] \end{array} \right\} \qquad \text{E6.3.6}$$

$\forall T_{cold\ i},\ i \in \left]2, n-1\right[:$

$$T_{cold\ i} = T_{cold\ in} + \frac{(W\cdot c_p)_{hot}}{(W\cdot c_p)_{cold}} \cdot \left(T_{hot\ i} - T_{hot\ in}\right) \qquad \text{E6.3.7}$$

Parallel flow heat exchanger

For parallel flow the procedure is slightly different. Initially, the temperatures of the hot and the cold flow at the first node are set to $T_{cold\ 1} = T_{cold\ in}$ and the $T_{hot\ 1} = T_{hot\ in}$ (figure 6.12). Then the temperatures for all remaining nodes are calculated:

$$\forall T_{cold\ i},\ i \in]2, n[:$$

$$T_{cold\ i} = \frac{1}{1 + \frac{(W \cdot c_p)_{cold}}{(W \cdot c_p)_{hot}}} \cdot \left\{ \begin{array}{l} T_{hot\ in}\left[1 - e^{-\left[\frac{1}{(W \cdot c_p)_{cold}} + \frac{1}{(W \cdot c_p)_{hot}}\right] \cdot UA \frac{(i-1)}{n}}\right] \\ + T_{cold\ in}\left[\frac{(W \cdot c_p)_{cold}}{(W \cdot c_p)_{hot}} + e^{-\left[\frac{1}{(W \cdot c_p)_{cold}} + \frac{1}{(W \cdot c_p)_{hot}}\right] \cdot UA \frac{(i-1)}{n}}\right] \end{array} \right\} \quad \text{E6.3.8}$$

$$\forall T_{hot\ i},\ i \in]2, n[:$$

$$T_{hot\ i} = T_{hot\ in} - \frac{(W \cdot c_p)_{cold}}{(W \cdot c_p)_{hot}} \cdot (T_{cold\ i} - T_{cold\ in}) \quad \text{E6.3.9}$$

The outlet pressure:

$$p_{cold\ out} = (1 - p_{cold\ loss}) \cdot p_{cold\ in} \quad \text{E6.3.10}$$

$$p_{cold\ out} = (1 - p_{cold\ loss}) \cdot p_{cold\ in} \quad \text{E6.3.11}$$

The shown procedure allows evaluating temperature pattern for the whole heat exchanger model at the design point.

6.3.2 The Sizing Model

If an off-design or transient study is followed after the design point, the sizing algorithm should follow the heat exchanger design point model. It determines the basic dimensions of the passages, which are used in heat transfer coefficient calculations. Thermodynamic properties at the inlet are delivered by Turbomatch and the temperature pattern of the model is known from the design point model.

First, the value of arithmetic mean pressures and logarithmic mean temperatures of the fluid are calculated from the inlet and outlet values of fluid pressure and temperature. These are necessary for further calculations. The specific gas constant R,

and heat capacity ratios γ are calculated by Turbomatch subroutine. Knowing the design Mach number, the static values of mean pressures and temperatures are calculated by E2.4.8. Therefore any quantity with and index *s mean* is supposed to be static. Other heat exchanger properties are calculated as follows:

The velocity of the fluid flow:

$$V_{cold\,(hot)} = \left(M \cdot \sqrt{\gamma\, R\, T_{s\ mean}}\right)_{cold\,(hot)} \qquad \text{E6.3.12}$$

The density of the fluid flow:

$$\rho_{cold\,(hot)} = \left(\frac{p_{s\ mean}}{R \cdot T_{s\ mean}}\right)_{cold\,(hot)} \qquad \text{E6.3.13}$$

The cross sectional area of the passage:

$$A_{cold\,(hot)} = \left(\frac{W}{\rho \cdot V}\right)_{cold\,(hot)} \qquad \text{E6.3.14}$$

The fluid conductivity:

$$k_{cold\,(hot)} = \left(\begin{matrix} 1.5207 \cdot 10^{-11} \cdot T_{s\ mean}^{3} - 4.8574 \cdot 10^{-8} \cdot T_{s\ mean}^{2} \\ +1.0184 \cdot 10^{-4} \cdot T_{s\ mean} - 3.9333 \cdot 10^{-4} \end{matrix}\right)_{cold\,(hot)} \qquad \text{E6.3.15}$$

The fluid dynamic viscosity:

$$\mu_{cold\,(hot)} = \left[1.827 \cdot 10^{-5} \cdot \frac{410.859}{120 + T_{s\ mean}} \cdot \left(\frac{1.8 \cdot T_{s\ mean}}{524.07}\right)^{\frac{3}{2}}\right]_{cold\,(hot)} \qquad \text{E6.3.16}$$

The fluid kinematic viscosity:

$$\nu_{cold\,(hot)} = \left(\frac{\mu}{\rho}\right)_{cold\,(hot)} \qquad \text{E6.3.17}$$

The height of the hot passage:

$$H_{hot} = \sqrt{\frac{A_{hot}}{AR}} \qquad \text{E6.3.18}$$

The contact area width:

$$Y = H_{hot} \cdot AR \qquad \text{E6.3.19}$$

The height of the cold passage:

$$H_{cold} = \frac{A_{cold}}{Y} \qquad \text{E6.3.20}$$

The passage hydraulic diameter:

$$\phi_{H\ cold(hot)} = \frac{2 \cdot H_{cold(hot)} \cdot Y}{H_{cold(hot)} + Y} \qquad \text{E6.3.21}$$

The fluid Prandtl number:

$$\mathrm{Pr}_{cold(hot)} = \left(\frac{c_p \cdot \mu}{k} \right)_{cold(hot)} \qquad \text{E6.3.22}$$

The Reinolds number:

$$\mathrm{Re}_{cold(hot)} = \left(\frac{V \cdot \phi_H}{\nu} \right)_{cold(hot)} \qquad \text{E6.3.23}$$

The fluid friction factor:

$$f_{cold(hot)} = \left[0.79 \cdot \ln\left(\mathrm{Re}_{cold(hot)} \right) - 1.64 \right]^{-2} \qquad \text{E6.3.24}$$

The Nusselt number of the fluid:

$$Nu_{cold(hot)} = \frac{f_{cold(hot)}}{8} \cdot \frac{\left(\mathrm{Re}_{cold(hot)} - 1000 \right) \cdot \mathrm{Pr}_{cold(hot)}}{1 + 12.7 \cdot \left(\frac{f_{cold(hot)}}{8} \right)^{0.5} \cdot \left(\mathrm{Pr}_{cold(hot)}^{\frac{2}{3}} - 1 \right)} \qquad \text{E6.3.25}$$

Heat transfer coefficient:

$$h_{cold(hot)} = \left(\frac{k \cdot Nu \cdot \chi}{\phi_H} \right) \qquad \text{E6.3.26}$$

χ is the convection increase factor. Its value is roughly determined as:

χ = 3 (for a recuperator)

χ = 1 (for an intercooler)

Overall heat transfer coefficient:

$$U = \frac{1}{\frac{1}{h_{cold}} + \frac{1}{h_{hot}}} \qquad \text{E6.3.27}$$

The length of the heat exchanger (the length of passages):

$$L = \frac{UA_{DP}}{U \cdot Y} \qquad \text{E6.3.28}$$

The main outcome of sizing model is the set of three dimensions of the heat exchanger model that are used in subsequent simulations.

6.3.3 Off-design Model

This model is applied for the gas turbine heat exchanger when the steady state off-design conditions are simulated. The dimensions of heat exchanger passages have been determined in the sizing algorithm. Two versions of the algorithm have been proposed in (Pellicer 2007, Pastor 2007), one faster with constant value of UA and second with variable UA, calculated from the new thermodynamic conditions in the heat exchanger. The second version of off-design model is a little bit slower, because it involves an internal matching loop where the outlet temperature of the colder stream is guessed and compared with the calculated one at the outlet. The off-design heat exchanger model has been ultimately used in Turbomatch, because it offers more precise UA calculation for only little costs of calculation time.

The thermodynamic parameters are once again provided by Turbomatch. Arithmetic mean values of fluid static pressures are obtained initially. Logarithmic mean of fluid temperatures cannot be determined at this point, because the outlet temperatures of the fluid are unknown. The value of the outlet temperature for the colder fluid is guessed. The outlet temperature of the hotter flow is determined from:

Counter flow heat exchanger:

$$T_{hot\,n+1} = T_{hot\,out} = \frac{1}{1-\dfrac{(W\cdot c_p)_{hot}}{(W\cdot c_p)_{cold}}} \cdot \left\{ \begin{array}{l} T_{hot\,in} \cdot \left\{ e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right] UA} - \dfrac{(W\cdot c_p)_{hot}}{(W\cdot c_p)_{cold}} \right\} \\ + T_{cold\,out} \cdot \left\{ 1 - e^{\left[\frac{1}{(W\cdot c_p)_{cold}} - \frac{1}{(W\cdot c_p)_{hot}}\right] UA} \right\} \end{array} \right\} \qquad \text{E6.3.29}$$

Parallel flow heat exchanger:

$$T_{hot\,n+1} = T_{hot\,out} = T_{hot\,in} - \frac{(W\cdot c_p)_{cold}}{(W\cdot c_p)_{hot}} \cdot \left(T_{cold\,out} - T_{cold\,in}\right) \qquad \text{E6.3.30}$$

Now the temperatures at the inlet and outlet of all passages are known. Their logarithmic mean static value is evaluated from which the variable gas properties are determined. To calculate the new values of heat transfer coefficients equations E6.3.12 – W6.3.27 are used. The new value of UA is determined from overall heat transfer coefficient (E6.3.27) heat exchanger dimensions:

$$UA = U \cdot Y \cdot L \qquad \text{E6.3.31}$$

The number of transfer units for off-design case:

$$NTU = \frac{UA}{(W\cdot c_p)_{\min}} \qquad \text{E6.3.32}$$

From which the new heat exchanger effectiveness for the off design case can be calculated knowing NTU and CR from E:

Counter flow heat exchanger:

$$\varepsilon_{HX} = \frac{1-e^{-NTU\cdot(1-CR)}}{1-CR\cdot e^{-NTU\cdot(1-CR)}} \qquad \text{E6.3.33}$$

Parallel flow heat exchanger:

$$\varepsilon_{HX} = \frac{1-e^{-NTU\cdot(1+CR)}}{1+CR} \qquad \text{E6.3.34}$$

The temperature distribution for all heat exchanger nodes (figure 6.11 and 6.12) is determined from E6.3.5 – E6.3.9. The calculated value of the outlet temperature of the cold stream is compared to the guessed value. If the difference between them exceeds certain tolerance, the calculated value becomes a new guess, heat transfer coefficients are updated and the new temperature distribution is determined.

6.3.4 The Transient Heat Exchanger Model

During transient gas turbine engine simulation the parameters at the inlet and outlet vary with time, and so do the thermodynamic parameters across the heat exchanger (Pellicer 2007, Pastor 2007). The relationship of thermodynamic parameters at the theoretical infinitesimally small segment dx is shown on figure 6.13.

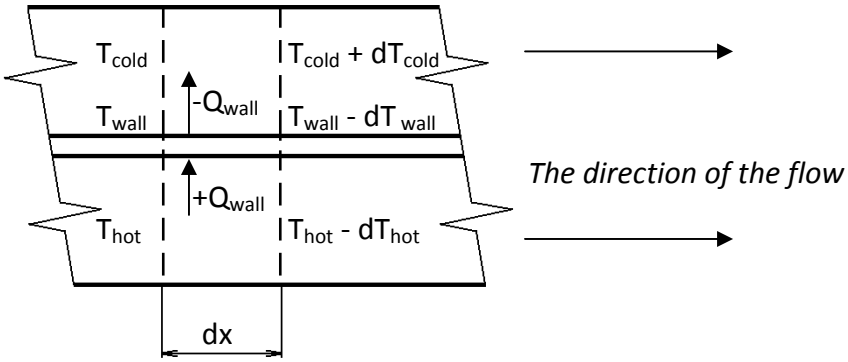


Figure 6.13: The thermodynamic parameters of an infinitesimally small heat exchanger model segment

Within this segment the temperature of the wall will be affected by the heat the wall receives from the hotter fluid and heat which will be released to the colder fluid. The logic is the same for counter and parallel flow heat exchanger with a small difference: The T_{cold} will be on the right side of the segment and the $T_{cold} + dT_{cold}$ will be on the left side.

The heat transferred to the wall from the hotter fluid:

$$+Q_{wall} = h_{hot} \cdot Y \cdot (dT_{hot}) \cdot dx \qquad \text{E6.3.35}$$

$$-Q_{wall} = h_{cold} \cdot Y \cdot (dT_{cold}) \cdot dx \qquad \text{E6.3.36}$$

The heat flux across the wall can be expressed as:

$$\frac{dU_{wall}}{dt} = \rho_{wall} \cdot Y_{wall} \cdot s_{wall} \cdot c_{p\,wall} \cdot \frac{dT_{wall}}{dt} dx \qquad \text{E6.3.37}$$

The temperature derivative is due to numerical purposes expressed as a temperature difference between transient time step:

$$\frac{dT_{wall}}{dT} = \frac{T^{t+dt} - T^{t}}{\Delta t} \qquad \text{E6.3.38}$$

After few arithmetic transpositions the correlations for transient variation of fluid and wall temperature is obtained. During the temperature calculations, the temperature of the wall is evaluated first, followed by fluid temperature calculations. The temperature correlations for actual transient time step are shown.

For counter flow heat exchanger:

$$T^{t}_{wall\,i} = T^{t-\Delta t}_{wall\,i} + \left(\frac{1}{\rho \cdot s \cdot c_p}\right)_{wall} \cdot \left[h_{hot} T^{t-\Delta t}_{hot\,i} + h_{cold} T^{t-\Delta t}_{cold\,i} - \left(h_{hot} + h_{cold}\right) \cdot T^{t-\Delta t}_{wall\,i}\right] \cdot \Delta t \qquad \text{E6.3.39}$$

$$T^{t}_{hot\,i} = \frac{1}{\frac{2 \cdot \left(W_{in} \cdot c_p\right)_{hot}}{h_{hot} \frac{L}{n} Y} + 1} \cdot \left[T^{t}_{hot\,i-1} \cdot \left(\frac{2 \cdot \left(W_{in} \cdot c_p\right)_{hot}}{h_{hot} \frac{L}{n} Y} + 1 \right) + T^{t}_{wall\,i} + T^{t}_{wall\,i-1} \right] \qquad \text{E6.3.40}$$

$$T^{t}_{cold\,i} = \frac{1}{\frac{2 \cdot \left(W_{in} \cdot c_p\right)_{cold}}{h_{cold} \frac{L}{n} Y} + 1} \cdot \left[T^{t}_{cold\,i+1} \cdot \left(\frac{2 \cdot \left(W_{in} \cdot c_p\right)_{cold}}{h_{cold} \frac{L}{n} Y} + 1 \right) + T^{t}_{wall\,i+1} + T^{t}_{wall\,i} \right] \qquad \text{E6.3.41}$$

For parallel flow heat exchanger:

For the temperature distribution in the wall and in the hot stream the correlations E6.3.39 and E6.3.40 are used. The temperature distribution in the cold stream is calculated as follows:

$$T^{t}_{cold\ i} = \frac{1}{\frac{2\cdot\left(W_{in}\cdot c_p\right)_{cold}}{h_{cold}\frac{L}{n}Y}+1}\cdot\left[T^{t}_{cold\ i-1}\cdot\left(\frac{2\cdot\left(W_{in}\cdot c_p\right)_{cold}}{h_{cold}\frac{L}{n}Y}+1\right)+T^{t}_{wall\ i}+T^{t}_{wall\ i-1}\right] \qquad \text{E6.3.41}$$

The temperature at the outlet corresponds to the temperature of the last stream node.

What the transient model proposed in (Pellicer 2007, Pastor 2007) does not account with is the volume packing and heat storage in the heat exchanger. Volumes of heat exchanger are usually quite large, hence their effect on overall engine transient performance is significant. The same applies for the heat storage. The proposed transient heat exchanger model accounts with the heat storage in the separating wall, but the storage of the heat in the fluid is not captured well. A solution to this was the addition of a control volume (chapter 3.1.2 and 3.1.3) after the heat exchanger model to model the volume packing and heat storage effects.

6.4 In-built Heat Exchanger Model Testing

In order to test the heat exchanger model, two engine models were created. Both have the same specification, the size and the same component, apart from the recuperator. appendix 6.1 shows the ICR engine model input file for the Turbomatch. The input file for the intercooled-recuperated engine is approximately the same, only the recuperator bricks are omitted (HEXCOL and HEXHOT).

Engine model with an Intercooler

First engine simulated was the two-spool turboshaft with power turbine and intercooler (figure 6.14)

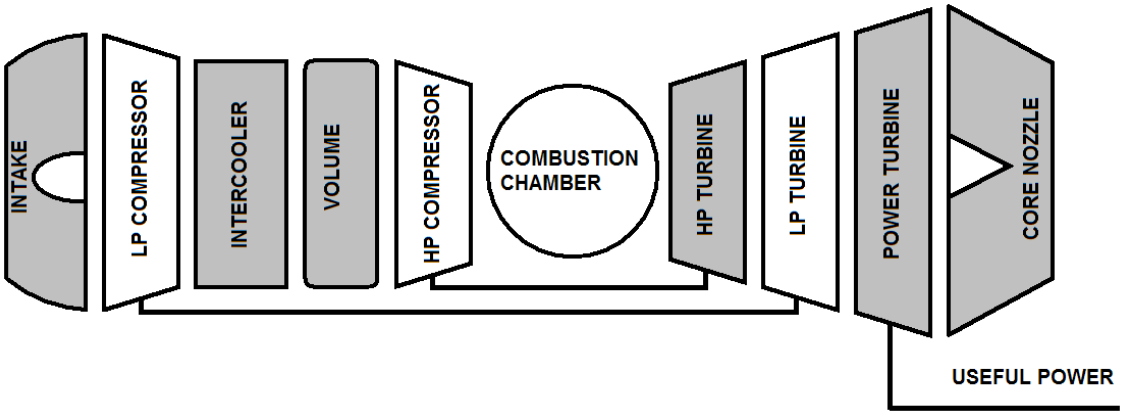


Figure 6.14: The simplified schema of the two spool gas turbine engine with an intercooler and a power turbine

For transient acceleration simulation the engine was set to accelerate abruptly from combustor outlet temperature of 1350 K to 1450 K. The control volume was only located at the end of the intercooler. Therefore the only engine components affected by the involved control volume is the LP compressor and the intake. Looking at figure 6.15 it is possible to observe a smooth compressor running for the LP compressor, whereas the HP compressor running lines begins with a steep sudden increase in pressure ratio along the speed line, and then it abruptly "turns" right to follow the increase in rotational speed due to shaft power imbalance.

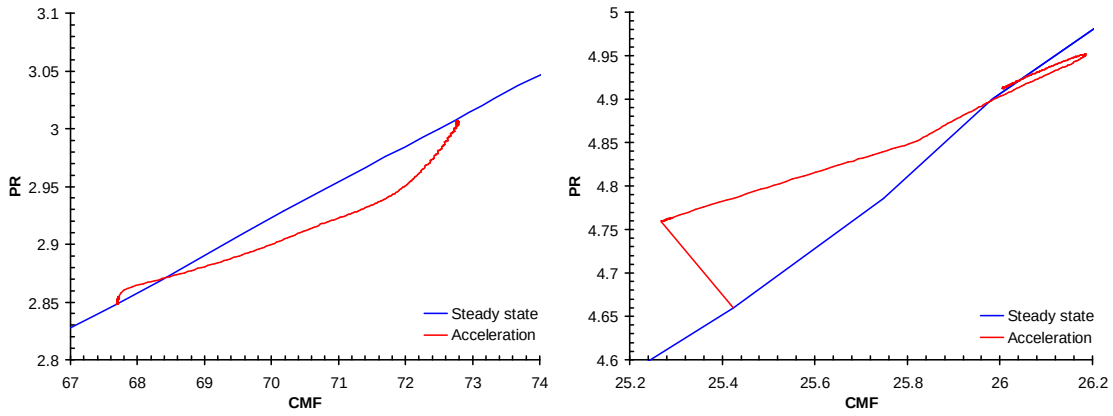


Figure 6.15: The steady state and the transient running line of the LP compressor (left) and the HP compressor (right)

The power turbine is assumed to have a fixed shaft speed. Figure 6.16 shows the variation of relative rotational speeds for the HP and LP shaft.

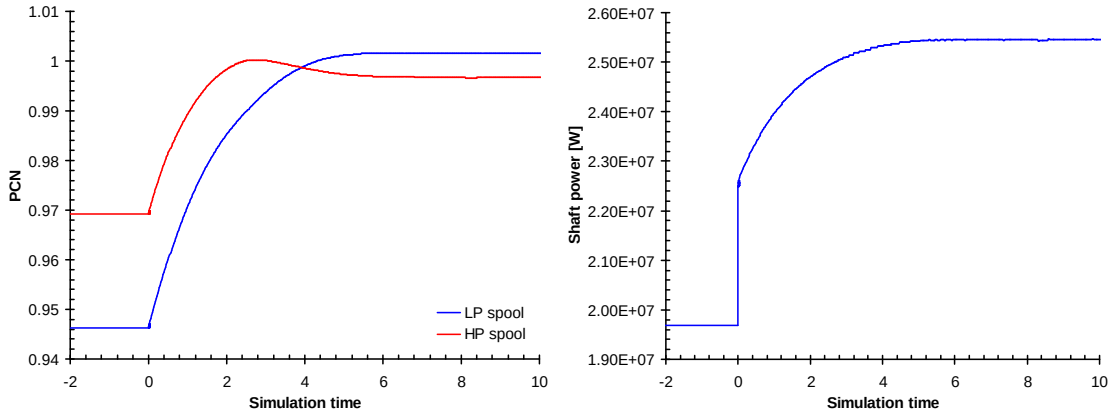


Figure 6.16: The transient behavior of the engine model with an intercooler: Variation of relative rotational speed in time (left), variation of the delivered shaft power (right)

Engine model with ICR cycle

A recuperator has been added to the original engine model for the second test (figure 6.17). Due to low pressure of the gas at the exhaust, the volume after the hot heat exchanger passage (the second recuperator in the engine model on figure 6.17) was very large – more than 6 m^3. The engine model is an interpretation of a marine gas turbine, where the dimensions for such big heat exchangers are acceptable. However if the engine was used for aero applications, heat exchanger size would be a very important parameter to watch.

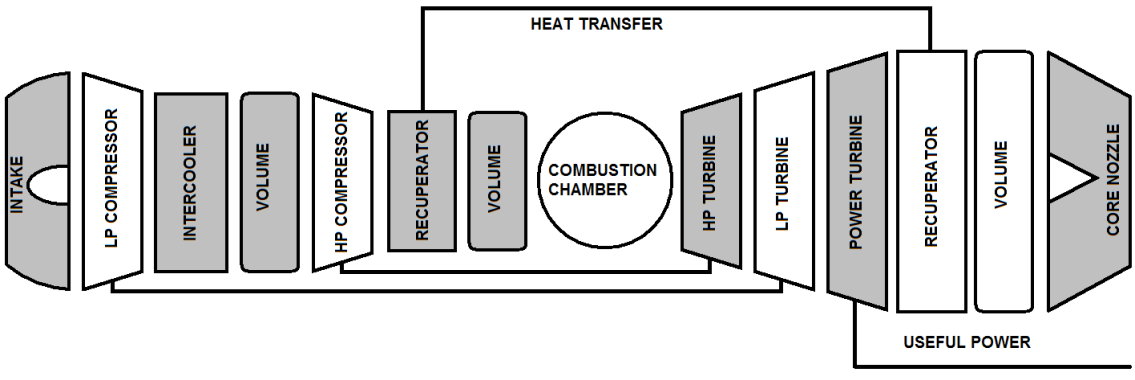


Figure 6.17: The simplified schema of the two spool gas turbine engine with an intercooler, recuperator and a power turbine

The steady state off-design analysis has been simulated at the range of temperatures between 1150K and 1600 K. There have been no complications. But the model is very unstable for any transient analysis. Instead of acceleration the engine decelerates when a sudden increase of combustor outlet temperature is set due decrease of the turbine power caused by the decrease in overall mass flow.

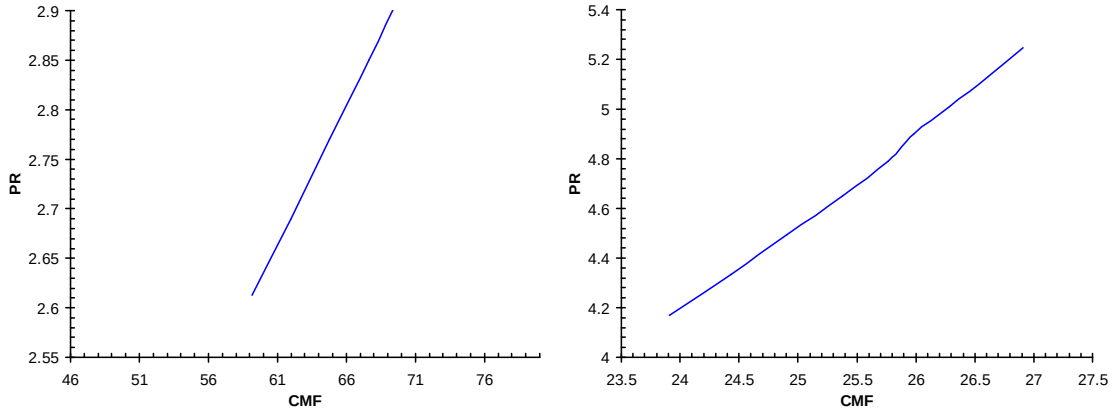


Figure 6.18: The steady state running line of the LP compressor (left) and the HP compressor (right)

CHAPTER 7

CONCLUSION

The research of transient performance modeling for engine flight path cycle analysis has been presented here. Turbomatch, the program of Cranfield University to simulate the engine performance behavior and to predict the engine thermodynamic cycle, was intended to be used for this purpose. A comparison of two major transient methods has been presented:

- Rapid transient performance method
- Thermodynamic matching transient method

The first method is used mainly for real-time engine simulation. But since Turbomatch uses number of internal loops to increase the engine modeling accuracy, the method performed in real time only for simplest engine configurations. The method has not been finally adopted

Thermodynamic matching method showed many advantages compared to the first method:

- Increased computational stability
- No parameter oscillations
- Fewer changes in Turbomatch structure needed
- Higher accuracy of simulation results
- In some cases faster due to longer permissible time step

In order to calculate the engine acceleration, implicit Euler integration of the time step has been used. The adopted method was tested on eleven gas turbine engines with various configurations:

- Basic turbojet
- Turbojet with combustion chamber diffuser
- Turbojet with convergent-divergent exhaust nozzle
- Gas generator with power turbine
- Basic two-spool configuration without bypassing flow
- Engine model of CFM56-7B interpretation
- Engine model of GE90-90B interpretation
- Engine model of GE 90-85B interpretation
- Two-spool gas turbine engine model with intercooled-recuperated cycle
- Three-spool turbofan with separated exhausts
- Engine model of RR RB211 interpretation with mixed exhaust

The results of transient simulation for four selected engine models have been presented. Charts showing the compressor working line excursion were obtained for all compressors. During the research an importance of virtual fuel schedule control system arose. Based on referred fuel flow vs. referred speed, the control system enables smooth fuel schedule based on user preferences. A quick start Turbomatch manual has been created. The engine model generation was shown with information how to set the fuel control system and how to obtain typical transient performance charts.

The methodology of engine flight path analysis (FPA) has been presented. Using developed tools user is able to simulate the variation of engine thermodynamic parameters across the whole flight. The main objective of FPA is to produce the thermodynamic parameters for engine life estimation and shop visit prediction tool.

The effect of various flight-dependent parameters on engine component life can be studied. The FPA may be run in steady state or transient mode. Steady state mode is faster to analyze, but it does not capture the engine dynamics during sensitive flight path segments, such as take-off.

At last, the model to simulate advanced gas turbine engines with intercooled recuperated technology has been developed, implemented and tested. The model heat transfer parameters are calculated by empirical correlations, on the contrary to original heat exchanger model. Two hypothetical engine models representing possible marine gas turbine configuration have been created. Because the heat exchanger model uses new algorithms for steady state operation as well, the convergence capabilities of the method have been tested on steady state first. The tests were a success for both models. The transient simulation for the gas turbine with intercooler only was also successful. Compressor working line excursion from steady state line has been shown.

CHAPTER 8

FUTURE RESEARCH PERSPECTIVE

The transient performance model has been tested on a range of gas turbine engines. The model is able to predict engine dynamic behavior following any change in ambient conditions dependent, or control system dependent parameters. Many interesting aspects during the research were observed, which could be subjected to further work.

8.1 Language Selection

The original version of Turbomatch was written in Fortran 77 and later rewritten to Fortran 90. Fortran 90 is a high-level programming language developed for numerical scientific and technological applications (Ellis 1994). It should be mentioned that there are already newer versions of Fortran, the latest being Fortran 2003 which emphasizes the interoperability with C language and deletes obsolete features used in Fortran 77 (Metcalf, Reid & Cohen 2004). This restriction however made Fortran 2003 inconvenient to use for the project because the compatibility with older version is required. Fortran 90 also has simple syntax programming and fast computation time.

The main drawbacks of Fortran used in Turbomatch included:

- Low default number of digits for declared REAL variables on a 32 bit computer environment. This problem was overcome by manual assignation of variables accuracy in subroutine PortSet.
- No object-oriented programming. Although Visual Fortran compilers have the ability to call subroutines from dynamic libraries for example.
- Fewer possibilities to apply user-friendlier GUI. Visual Fortran has the ability to use some GUI features, but in order to do so in Windows for example, programmer needs to use Windows API routines and should be comfortable with writing C

application and be familiarized with .NET Framework SDK. Currently Pythia written in Visual Basic provides a GUI for Turbomatch including additional features such as runtime component degradation control. However there were issues reported resulting from working in mixed-language environment

In future Turbomatch could be rewritten to C++ which possesses high degree of CPU control and OOP (Schildt 2003.). Mathematical functions are not intrinsic but a math library can be called where default floating-point variables used are stored on two-times more bites compared do REAL variables in Fortran which results in higher accuracy (Longer time needed for such calculation can be neglected on modern computers). The drawback high degree of CPU control is that the code is a little bit more inclinable to human errors than Fortran that makes C++ an unsafe language. An alternative could be also C# programming language which is safe in this way and its main advantage is portability and rapid application development. C# language requires NET Framework to be installed on Windows or Mono on Unix-based operation systems (Linux or Mac OS). The code is not compiled into native code but MSIL (Microsoft Intermediate language) and then compiled with JIT (Just-In-Time) compiler of NET Framework (Schildt 2001). Simulation programs written in C# have much longer runtime (up to 10 – 20 times) compared to Fortran or C++ programs. But because of high degree of OOP and support in MS Visual Studio, the development time of C# application is shorter than development time of application with the same algorithm written C++. The decision of what language should be used in future development will be also influenced by fact that computers are becoming faster and programmers working hours more expensive.

8.2 Fuel Control Algorithm

The fuel schedule is during the transient operation a very important task for the control system. The system developed in this thesis is based on basic assumptions and it monitors only rotational speed and combustor outlet temperature to schedule the fuel. Modern fuel control systems in aircraft systems determine the acceleration rate, instead of the fuel flow. The fuel flow is calculated by the engine model accordingly. Such control system is often more reliable for atypical engine operating conditions, moreover it provides easier engine control with respect to user.

8.3 Deeper Insight into Recuperator Transient Model

The heat exchanger model implemented to Turbomatch showed great stability for the design and off-design calculations. The transient simulations carried out for a gas turbine with an intercooler showed stability across the whole transient acceleration simulation. After a recuperator was added, the engine started to decelerate instead of acceleration, when the combustor outlet temperature was increased. Further research into recuperator transient model is suggested.

8.4 Dynamic Interconnection of Flight parameters with transient algorithm

The way how transient simulation is modeled does not permit the variation of flight dependent parameters during the simulation. It is only possible to feed the variations of the parameters at the beginning of the engine simulation. It would be very useful for FPA if the parameters were updated according to the flight evolution. For example, during a take-off the variation of Mach number and altitude with time depends on the engine thrust and aircraft lift and drag polar. Current model requires the variation of Mach number, altitude and other parameters to be delivered at the beginning of the simulation. An interconnection between Hermes and Turbomatch may ensure that the parameters will be provided to transient algorithm just-in-time according to the flight evolution.

REFERENCES

Abbasov, N.M. Zeinalov, R.I., Azizova, O.M. & Imranova, S.N. (2006). "Dynamic models of heat exchangers (translated)". *Chemistry and Technology of Fuels and Oils,* vol. 42, no. 1, pp. 25-29.

Beyer, W.H. (1981). *CRC standard mathematical tables.* 26th edn, Chemical R.C. Press, Boca Raton, Florida.

Boggia, R. (2005). "Intercooled recuperated gas turbine engine concept". 41st AIAA/ASME/SAE/ASEE join propulsion conference & exhibit, Tuscon, Arizona.

Bücker, D. Span, R. & Wagner, W. (2003). "Thermodynamic property models for moist air and combustion gases". *Journal of engineering for gas turbines and Power,* vol. 125, no. 1.

Cardarelli, F. & Shields, M.J. (1997). *Scientific unit conversion : a practical guide to metrication.* Springer-Verlag.

Chappell, M.S. Cockshutt, E.P. & Sharp, C.R. (1964). *Gas turbine cycle calculations: design point performance of turbojet and turbofan engines*. National Research Council of Canada, Aeronautical Report, LR-407, Canada.

Chilvers, I.M. & Milanović, J.V. (2002). "Transient analysis of a combined cycle power plant (CCGT) connected directly to the distribution network", IEE Conference Publication, pp. 461-466

Cockshutt, E.P. & Sharp, C.R. (1965). *Gas turbine cycle calculations: design point performance of turbopropellor and turboshaft engines*. National Research Council of Canada, Aeronautical Report, LR-448, Canada.

Da Cunha Alves, M.A. et al. (2001). "An insight on intercooling and reheat gas turbine cycles". *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy,* vol. 215, no. 2, pp. 163-172.

Dahlquist, G. Björck, Å. (1974). *Numerical methods by Germund Dahlquist and Åke Björck. Translated by Ned Anderson*.

Demoulin, S.C.L. (2004). *Derivation of intermediate compressor maps in order to improve performance calculations.* Cranfield Engineering, School of Mechanical Engineering, MSc thesis.

Ellis, T.M.R. (1994). *Fortran 90 programming.* Addison-Wesley, Wokingham.

Escher, P.C. (1995). *Pythia : an object-oriented gas path analysis computer program for general applications.* Cranfield University.

FAA (2003). *CFM56-7B Type certificate data sheet E00055EN*. U.S. Department of Transportation, Federal Aviation Administration.

Fawke, A.J. & Saravanamuttoo, H.I.H. (1971). "Digital computer methods for prediction of gas turbine dynamic response", *Paper 710550*.

Fishbach, L.W. & Koenig, R.W. (1972). *GENENG II - A program for calculating design and off design performance of two and three-spool turbofans with as many nozzles*, TND-6553, NASA.

Giordano, D. et al. (1994). *Tables of internal partition functions and thermodynamic properties of high-temperature air species from 50K to 100,000K.* European Space Agency, Paris.

Goudou, P. (2003). *Certification specification for large airplanes CS-25.* 2003/2/RM European aviation safety agency, Brussels.

Goudou, P. (2003). *Certification specitications for normal, utility, aerobatic and commuter category aeroplanes CS-23*. 2003/14/RM European aviation safety agency, Brussels.

GSP (2004). *GSP 10 User manual*. GPS Development team, National Aerospace Laboratory NLR, Anthony Fokkerweg 2, 1006 BM Amsterdam, The Netherlands.

Gulati, A. (2001). *An optimization tool for gas turbine engine diagnostics*. Cranfield University, PhD Thesis.

Gunston, B. & Jane's, I.G. (2004). *Jane's aero-engines.* Jane's Information Group, Coulsdon, Surrey, U.K. ; Alexandria, Va.

Hariharan, H. (2009). *Severity estimation and shop visit prediction of civil aircraft engines.* Cranfield University, School of Engineering, PhD Thesis.

ICAO (2007). *Engine exhaust emissions data bank (GE90-94B)*. ICAO.

ISO. (2009). *ISO 2533:1975 Standard Atmosphere.*

Jackson, P., Peacock, L. & Munson, K. (2006). *Jane's all the world's aircraft 2006-200.,* Janes Information Group.

Koenig, R.W. & Fishbach, L.W. (1972). *GENENG - A program for calculating design and off design performance for turbojet and turbofan engines.* TND-6555, NASA.

Koff, B.L. "Designing for fighter engine transients", *NATO AGARD Conference Proceedings No 324 - Engine Handling, AGARD, Brussel*.

Kong, C. K. et al. (2004). *Steady-state/transient performance simulation of the propulsion system for the canard rotor wing uav during flight mode transition.* Proceedings of the ASME Turbo Expo 2004.

Kong, C., Kho, S. & Ki, J. (2004). "Component map generation of a gas turbine using genetic algorithms". Proceedings of the ASME Turbo Expo 2004.

Kong, C., Ki, J. & Kang, M. (2004). "Fuzzy approaches for searching optimal component matching point in gas turbine performance simulation". *Journal of engineering for gas turbines and power,* vol. 126, no. 4.

Kurzke, J. (2005). "How to create a performance model of a gas turbine from a limited amount of information", Proceedings of the ASME Turbo Expo, 2005.

Kurzke, J. & Riegler, C. (2000). "A new compressor map scaling procedure for preliminary conceptional design of gas turbines". Proceedings of ASME IGTY, Turbo Expo 2000, Munich.

Kurzke, J. (1996). "How to get component maps for aircraft gas turbine performance calculations". American Society of Mechanical Engineers.

Lavelle, T.M. & Curlett, B.P. October (1994). *Graphical user interface for the NASA FLOPS aircraft performance and sizing code*. NASA, Lewis research center, Cleveland, Ohio 44135.

Lawley, J., Thompson, M. (2004). *Aircraft maintenance costs : a review of the third-party maintenance market.* Cranfield University, School of Engineering.

MacIsaac, B.D. & Saravanamuttoo, H.I.H. (1974). "A comparison of analog, digital and hybrid computing techiques for simulation of gas turbine performance". *Journal of engineering for gas turbines and Power,* vol. ASME 74-GT-127.

MacMillan, W.L. (1974). *Development of a modular type computer program for the calculation of gas turbine off design performance*, Cranfield Institute of Technology.

Maddalon, D.V. (1978). *Estimating airline operating costs.* Nasa, Washington, D.C.

Mattingly, J.D. (1996). *Elements of gas turbine propulsion.* McGraw-Hill, New York, London.

Mattingly, J.D. (2006). *Elements of propulsion: gas turbines and rocket.* American Institute of Aeronautics and Astronautics, Reston, VA.

McCullers, L.A. (1984). *Aircraft configuration optimizationincluding optimized flight profiles, multidisciplinary analysis and optimization Part 1*. NASA.

McDonald, C.F. et al. (2008). "Recuperated gas turbine aeroengines, part II: engine design studies following early development testing". *Aircraft Engineering and Aerospace Technology,* .

McKinney, J.S. (1967). *Simulation of turbofan engine, part 1. Description of method and balancing technique*. United States Air Force, Aero Propulsion Lab.

McKinney, J.S. (1967). *Simulation of turbofan engine, part 2. Users Manual and computer listing.* United States Air Force, Aero Propulsion Lab.

Metcalf, M., Reid, J. & Cohen, M. (2004). *Fortran 95/2003 explained.* Oxford University Press, Oxford.

NATO. (2007), *Performance prediction and simulation of gas turbine engine operation for aircraft, marine, vehicular and power generation.* Research and technology organisation, North athlantic threaty organisation, BP-25, F-92201 Neuilly-sur-Seine Cedex, France.

Neal, P.F. "Mechanical and thermal effects on the transient and steady state component performance of gas turbine engines.", *NATO AGARD Conference Proceedings No 324 - Engine Handling*.

Ogaji, S., Kyritsis, V. & Laskaridis, P. (2007). *Report on aircraft model development (Hermes) and user guide.* Cranfield University, School of Engineering.

Palmer, J.R. (1967). *The "Turbocode" scheme for the programming of thermodynamic cycle caclulations on an electronic digital computer.* College of Aeronautics Report.

Palmer, J.R. & Annand, K.P. (1968). *Description of the Algol version of the "Turbocode" scheme for programming of thermodynamic cycle calculations on an electronic digital computer.* College of Aeronautics Report.

Pastor, F.A. (2007). *Study and development of an algorithm for the transient performance response of a heat exchanger recuperator.* Cranfield University, School of Engineering.

Pellicer, A. (2007). *Transient performance model of a gas turbine intercooler. Cranfield University, School of Engineering*.

Pilidis, P. (2006). *Gas turbine theory and performance.* Course notes, School of Engineering, Cranfield University.

Pons, C. (2006). *Transient models of heat exchangers in a gas turbine*, School of Engineering, Cranfield University.

Press, W.H. & Teukolsky, S.A. (1994). *Numerical recipes in FORTRAN: the art of scientific computin.,* 2nd edn, Cambridge University Press, Cambridge.

Ragland, T.L. (1995). "High efficiency recuperated cycle, optimized for reliable, low cost, industrial gas turbine engines", American Society of Mechanical Engineers.

Reale, M. (2006). *New High Efficiency Simple Cycle Gas Turbine - GE's LMS100.* GE Energy.

Ridout, D. & McShane, B. (1988). *Numerical analysis, algorithms and computation.* Ellis Horwood, Chichester.

Riegler, C., Bauer, M. & Kurzke, J. (2001). "Some aspects of modelling compressor behavior in Gas turbine performance calculations". *Journal of Turbomachinery,* vol. 123, no. 2, pp. 372-378.

Riegler, C., Bauer, M. & Schulte, H. (2003). "Validation of a mixed flow turbofan performance model in the sub-idle operating range", American Society of Mechanical Engineers, International Gas Turbine Institute, Turbo Expo (Publication) IGTI.

Sanghi, V. Lakshmanan, B.K. & Rajasekaran, R. (2001). "Aerothermal model for real-time digital simulation of a mixed-flow turbofan engine". *Journal of Propulsion and Power,* vol. 17, no. 3, pp. 629-635.

Saravanamuttoo, H.I.H. (2001). *Gas turbine theory.* 5th edn, Prentice Hall, Harlow.

Saravanamuttoo, H.I.H. & Fawke, A.J. (1973). "Digital computer simulation of the dynamic response of a twin-spool turbofan with mixed exhausts". *Aeronautical Journal,* vol. 77, no. 753, pp. 471-478.

Schildt, H. (2001), *C# : a beginner's guide.* Osborne/McGraw-Hill, London.

Schildt, H. (2003)., *C++ : a beginner's guide.* 2nd edn, McGraw-Hill/Osborne ; McGraw-Hill, Emeryville, Calif. : London.

Siringlou, A.A. (1992). *Implementation of variable gometry for gas turbine performance simulation - Turbomatch improvement.* Cranfield Institute of Technology, School of Mechanical Engineering.

Tong, M.T. Halliwell, I. & Ghosn, L.J. (2004). *A computer code for gas turbine engine weight and disk life estimation.* Asme, New York, NY.

Torenbeek, E. (1982). *Synthesis of subsonic airplane design : an introduction to the preliminary design of subsonic general aviation and transport aircraft, with emphasis on layout, aerodynamic design, propulsion, and performance.* Delft University Press ; Kluwer Academic Publishers, Delft : Dordrecht.

U.S. standard atmosphere. (1976). Dept. of Commerce, National Oceanic and Atmospheric Administration

Van den Haut, F. (1991). *Gas turbine performance simulation: Improvements to the Turbomatch scheme*. Cranfield Institute of Technology, School of Mechanical Engineering, Depratment of PPA.

Walsh, P.P. & Philip, P. (2004). *Gas turbine performance,* 2nd edn, Blackwell Science, Oxford.

Wang, X. Su, S. & Lian, X. (2002). "Numerical simulation of turbofan engine with mixer afterburning under transient conditions". *Tuijin Jishu/Journal of Propulsion Technology,* vol. 23, no. 3, pp. 189-192.

Yadav, R. Kapadi, Y. & Pashilkar, A. (2005). "Aero-thermodynamic model for digital simulation of turbofan engine". Proceedings of the ASME Turbo Expo, 2005, pp. 63-70.

Yin, J., Hales, R., Pilidis, P., Curnock, B. & Meads, R. (2000). "Transient performance for high-bypass ratio turbofan models". *Aircraft Engineering and Aerospace Technology,* vol. 72, no. 6, pp. 518-523.

Zhidkov, N.P., Booth, A.D. & Berezin, I.S. (1965). *Computing methods,* Pergamon, Oxford.

The turbomatch scheme for aero/industrial gas turbine engine design point/off design performance calculation. 2007, School of Engineering, Cranfield University.

Arithmetic and logarithmic mean temperature difference. (2005). Available: http://www.engineeringtoolbox.com/arithmetic-logarithmic-mean-temperature-d_436.html [accessed: 2009, 07/11].

Aircraft Propulsion System Performance Station Designation and Nomenclature. (1997). , SAE International, Warrendale, Pennsylvania, USA.

B737 Detailed technical data. Available: http://www.b737.org.uk/techspecsdetailed.htm [accessed: 2009, 04/12].

Boeing - Commercial airplane reference guide 2007 - 2008, Boeing Commercial Airplanes, P.O. Box 3707, Seattle, WA 98124-2207 USA.

Boeing commercial airplanes. Available: http://www.boeing.com/commercial/ [accessed: 2009, 10/15].

*Cross flow heat exchanger*s. Available: http://faculty.kfupm.edu.sa/me/antar/experiments/Updated.Experiments/Cross_Flow/classes/index.htm [2009, 09/01].

*ICAO - Aviation statistics and airline data from ICAO Data. Air transport industry statistical data inc. traffic, personnel and financial information on airlines and airport*s. Available: http://icaodata.com/Terms.aspx#PassengerWeight [accessed: 2009, 10/15].

*The Weather channel: National and local weather forecast, hurricane, radar and report*s. Available: http://www.weather.com/ [accessed: 2009, 12/02].

World aeronautical database. Available: http://worldaerodata.com/ [accessed: 2009, 09/29].

APPENDIXES

Appendix 2.1 Examples on Source Code Clean-ups

Irrelevant comments in the code

These are few examples of comments deleted in Master_01_TURBOMAT.f90 as they were not considered as useful in the delivered code:

Programmers' communication

Line 427: "! HELPFLAG = 0.0 !... do you still want to retain this here now we have the new routine ManualInputSelection ?"

Line 600: "!... should we remove the INOT=0 from the code?"

Line 2066: "!... MESSY BIT to be reorganised"

Turned off code snippets

Line 210: "! REAL:: TMPFAC2" – example of variable that was no longer needed

Line 635: "! GOTO(35,35,40,40,45,45,50,50,55,55,60,60,62,62,64,64,65,65,& 70,70,75,75,77),J" – example of GOTO statement that was replaced by SELECT CASE block.

The demonstration of source code clarity improvement on INTAKE calling sequence

The code snippet of INTAKE calling sequence extracted from Turbomat of Turbomatch Intermediate version (90 lines of code):

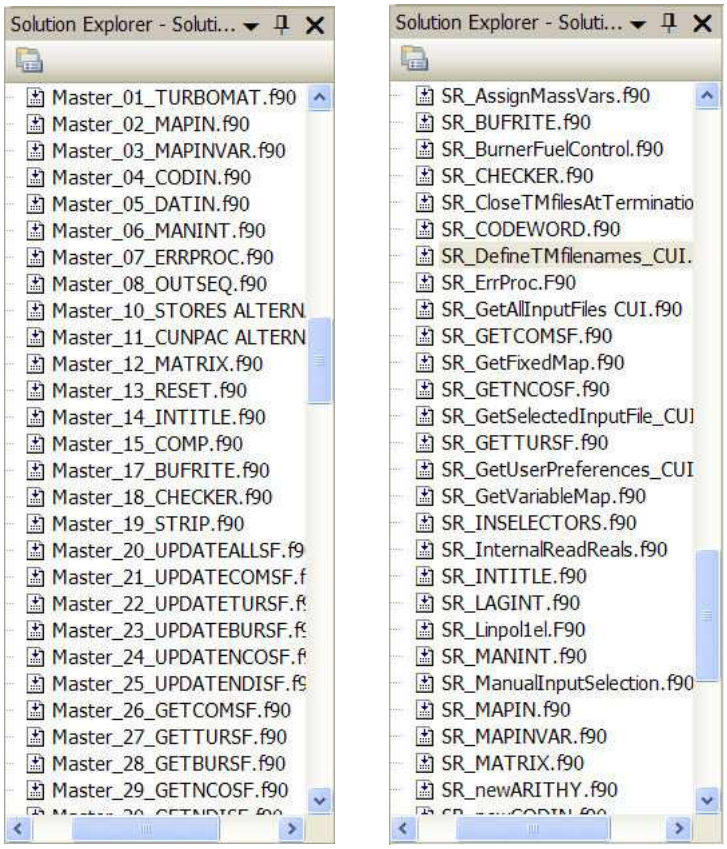
```
!     CHAPTER 11.1 INTAKE *******************************************

670   CONTINUE
    CALL CheckDataINTAKE(iBRICK,RETURN_STATUS)
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE bricktype1 bounds ',LBOUND(bricktype1,1),'  ( ',iBRICK,'
!) ',UBOUND(bricktype1,1)

    D(1)=D(1)/UNITSLENGTH
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE in R1=',R1

!RLHSTART introduce StationVector module
    CALL real_to_SV(SV(SA,1),SV(SA,2),SV(SA,3),SV(SA,4),&
                    SV(SA,5),SV(SA,6),SV(SA,7),SV(SA,8),SV(SA,9),SV(SA,10),SVa)
    CALL real_to_SV(SV(SB,1),SV(SB,2),SV(SB,3),SV(SB,4),&
                    SV(SB,5),SV(SB,6),SV(SB,7),SV(SB,8),SV(SB,9),SV(SB,10),SVb)
    CALL SVdifferences(SVa,Station(SA)%Vector)
```

```
    CALL SVdifferences(SVb,Station(SB)%Vector)
    ! Station(SA)%NextBrick%IBR should be 1 for INTAKE
    IF( Station(SA)%NextBrick%IBR/=1 )THEN
        !WRITE(13,*) 'DEBUG TURBOMAT INTAKE is wrong ',Station(SA)%NextBrick%IBR
    END IF
    IF( FINISHED/=FINISHED_ep )THEN
        !WRITE(13,*) 'DEBUG TURBOMAT INTAKE FINISHED,FINISHED_ep =',FINISHED,FINISHED_ep
    END IF
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE iBRICK =',iBRICK
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE D(1) =',bricktype1(iBRICK)%Altitude
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE D(2) =',bricktype1(iBRICK)%ISADev
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE D(3) =',bricktype1(iBRICK)%MachNo
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE D(4) =',bricktype1(iBRICK)%Precov
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE inlet station =',bricktype1(iBRICK)%Inlet%Number
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   F  =',bricktype1(iBRICK)%Inlet%Vector%F
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   W  =',bricktype1(iBRICK)%Inlet%Vector%W
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   PS =',bricktype1(iBRICK)%Inlet%Vector%PS
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   P  =',bricktype1(iBRICK)%Inlet%Vector%P
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   TS =',bricktype1(iBRICK)%Inlet%Vector%TS
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   T  =',bricktype1(iBRICK)%Inlet%Vector%T
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   V  =',bricktype1(iBRICK)%Inlet%Vector%V
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   A  =',bricktype1(iBRICK)%Inlet%Vector%A
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE called'
!      CALL INTAKE(DESignPointCase,FINISHED,IBR,NOWPRINT,D(1),D(2),D(3),D(4),R1,SB,&
!     SV(SA,1),SV(SA,2),SV(SA,3),SV(SA,4),SV(SA,5),SV(SA,6),SV(SA,7),&
!     SV(SB,1),SV(SB,2),SV(SB,3),SV(SB,4),SV(SB,5),SV(SB,6),SV(SB,7),&
!     SV(SB,8))
      CALL INTAKE(DESignPointCase,FINISHED,&
!     IBR,&
     NOWPRINT,&
     bricktype1(iBRICK),&           ! INTAKE object
     !D(1),D(2),D(3),D(4),&
     !R1,&
!     SB,&
     !SVa,&
     !SVb,&
     RETURN_STATUS1)
    IF( RETURN_STATUS1/=0 )THEN
      ! non-convergence in AIRTAB , AFROMV , VFROMA - Ignore all or part of this Point
      ! remedial action + error message with IBR,SB
      !WRITE(13,*) 'DEBUG TURBOMAT INTAKE return error',RETURN_STATUS1
      RETURN_STATUS=9716
      RETURN
    END IF
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE number',iBRICK,'   results:'
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE D(4) =',bricktype1(iBRICK)%Precov
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   R1 =',bricktype1(iBRICK)%Drag
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE outlet station =',bricktype1(iBRICK)%Outlet%Number
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   F  =',bricktype1(iBRICK)%Outlet%Vector%F
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   W  =',bricktype1(iBRICK)%Outlet%Vector%W
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   PS =',bricktype1(iBRICK)%Outlet%Vector%PS
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   P  =',bricktype1(iBRICK)%Outlet%Vector%P
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   TS =',bricktype1(iBRICK)%Outlet%Vector%TS
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   T  =',bricktype1(iBRICK)%Outlet%Vector%T
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   V  =',bricktype1(iBRICK)%Outlet%Vector%V
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE   A  =',bricktype1(iBRICK)%Outlet%Vector%A
    D(4)=bricktype1(iBRICK)%Precov
    R1  =bricktype1(iBRICK)%Drag
    !WRITE(13,*) 'DEBUG TURBOMAT INTAKE out R1=',R1
    SVa =bricktype1(iBRICK)%Inlet%Vector
    SVb =bricktype1(iBRICK)%Outlet%Vector
    CALL real_from_SV(SV(SA,1),SV(SA,2),SV(SA,3),SV(SA,4),&
                      SV(SA,5),SV(SA,6),SV(SA,7),SV(SA,8),SV(SA,9),SV(SA,10),SVa)
    CALL real_from_SV(SV(SB,1),SV(SB,2),SV(SB,3),SV(SB,4),&
                      SV(SB,5),SV(SB,6),SV(SB,7),SV(SB,8),SV(SB,9),SV(SB,10),SVb)
    Station(SA)%Vector=SVa
    Station(SB)%Vector=SVb
!RLHEND introduce StationVector module

    !IF( nBERRprint==1 )THEN
    !  WRITE(13,'("DEBUG TURBOMAT INTAKE - TRIGGER set")')
    !  TRIGGER=.TRUE.
    !END IF

!****ADDED BY F VAN DEN HOUT/CD NOZZLE EXIT AREA LIMITATION*********
      CDEAL=SV(SB,8)        ! this effectively a RESULT of the INTAKE calculation
      CDEAL=CDEAL*UNITSAREA     ! used as input to NOZDIV
                    ! there is now an easy way to get the value of INTAKE exit
! area directly from that brick
!****UNTIL HERE*****************************************************

!RLH  GOTO 945                 ! re-assign INNdex before jumping to 950
    INNdex=1
    GOTO 950
```

The code snippet of INTAKE calling sequence extracted from Turbomat subroutine of Turbomatch Transient version (29 lines of code):

```
!      CHAPTER 11.1 INTAKE
!=============================================================================================================
! Find:INTAKE

670   continue

    call CheckDataINTAKE(iBRICK,RETURN_STATUS)      ! This subroutine does not do anything
! yet (KEEP IT)

    call INTAKE(&
        DESignPointcase,&
        FINISHED_ep,&
        NOWPRINT,&
        bricktype1(iBRICK),&        ! INTAKE object
        MessagesAreRequired,&
        RETURN_STATUS1&
        )
    if ( RETURN_STATUS1/=0 ) then
        ! non-convergence in AIRTAB , AFROMV , VFROMA - Ignore all or part of this Point
        ! remedial action + error message with IBR,SB
        !write(13,*) 'DEBUG TURBOMAT INTAKE return error',RETURN_STATUS1
        RETURN_STATUS=9716
        return
    end if

!****ADDED BY F VAN DEN HOUT/CD NOZZLE EXIT AREA LIMITATION*********

            CDEAL = bricktype1(iBRICK)%Outlet%Vector%A
            CDEAL = CDEAL*UNITSAREA     ! used as input to NOZDIV
            ! there is now an easy way to get the value of INTAKE exit area directly
! from that brick
!****UNTIL HERE*****************************************************

    INNdex=1    ! Tells STORES subroutine where was it CALLED from
    goto 950
```

Appendix 2.2



An example of code files order in Legacy (left) and Transient (right) Turbomatch version

Appendix 2.3 Arithmetic Operation Precision Demonstration

As an example, the sum one smaller and one higher number will be evaluated. If these two numbers were e.g. 1/3 and 1/700 in 32-bit computer environment they will be stored as $0.333333x10^{0}$ and $0.142857x10^{-2}$ in computer memory. Arithmetic operations are carried out in registers where both numbers have more valid digits available. Numbers copied from the memory will be set to have the same exponent. After the operation the result is loaded into the memory where the number has fewer digits available for the precision, as compared in registers:

Registers: $0.333333x10^{0} + 0.00142857x10^{0} = 0.33476157x10^{0}$

Memory: $0.334761 \quad x10^{0}$

In this case the loss of precision is very small and it represents only 0.04% of second addend. However, if there was any convergence criterion which would require the error to be less then it would be not possible for the criterion to be met as the error will always oscillate around higher value because of the precision loss.

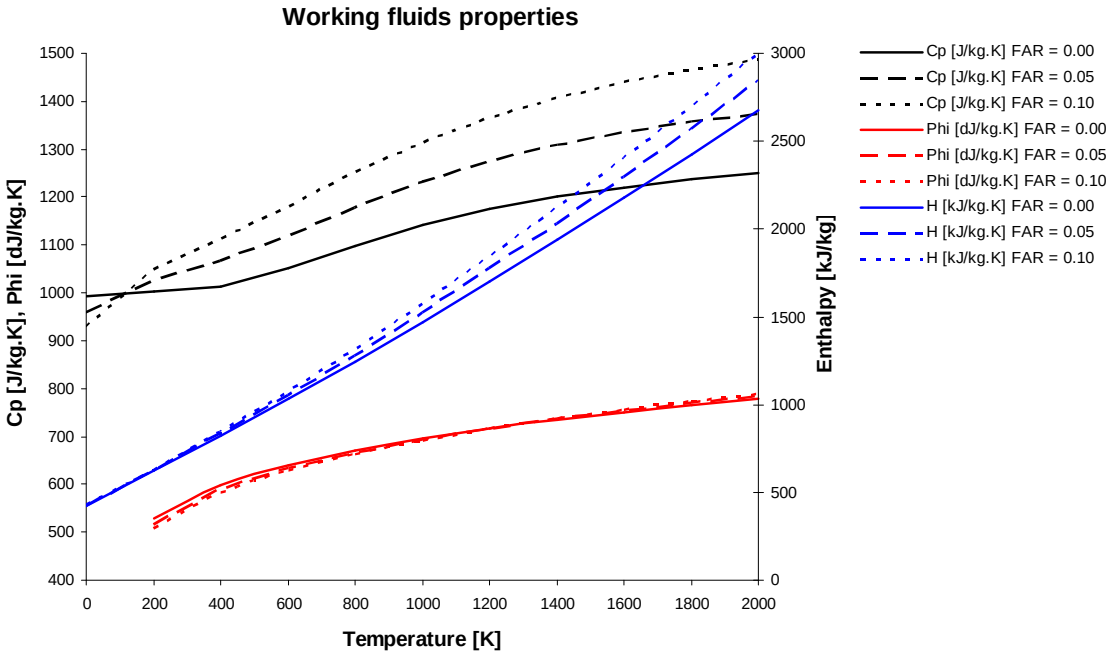
Even more serious problem could appear if someone would like to evaluate the sum of two number of which one would be so small that its effect on the final sum would be out of memory precision range, for example 1/3 and 1/2 000 000 ($0.333333x10^{0}$ and $0.50000x10^{-6}$):

Registers: $0.333333x10^{0} + 0.000000500000x10^{0} = \quad 0.3333335x10^{0}$

Memory: $0.333333 \ x10^{0}$

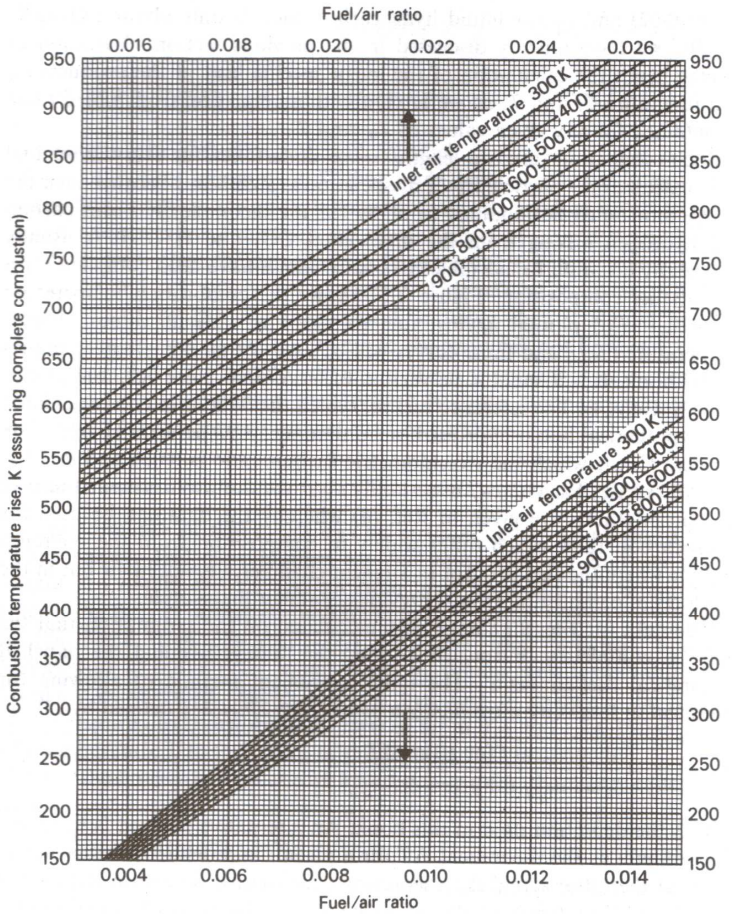
As seen above, the sum has the same value as the first addend.

Appendix 2.4



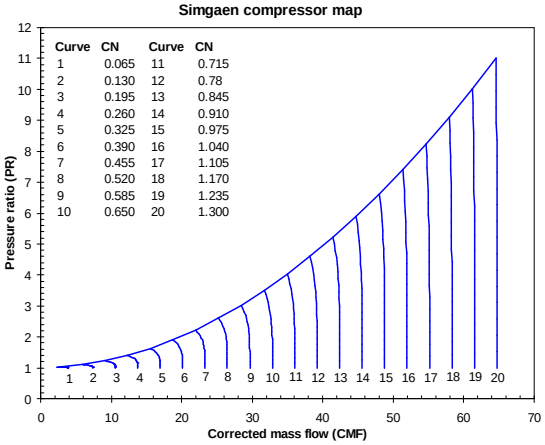
Working fluid properties are evaluated by polynomials obtained from (Walsh, Philip 2004) and the chart is just their visualization. In Turbomatch similar approach has been used, however working fluid polynomials are different compared to those used in Simgaen. Polynomials for Turbomatch intermediate and Transient version were taken from (Bücker, Span & Wagner 2003).

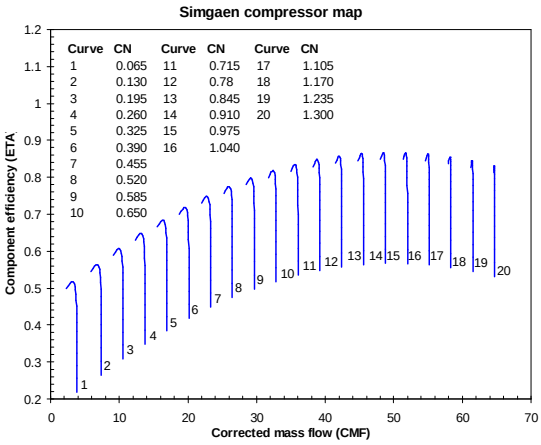
Appendix 2.5

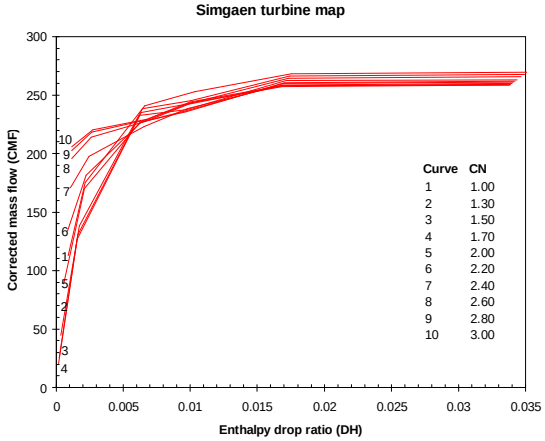


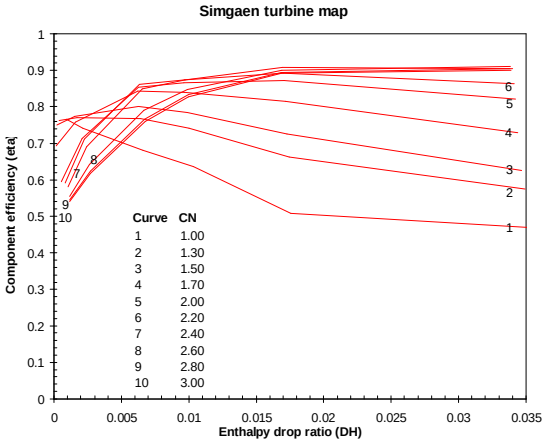
A chart showing combustion chamber temperature rise vs. FAR (ref. (Walsh, Philip 2004))

Appendix 2.6 Compressor and Turbine Maps Used in Simgaen

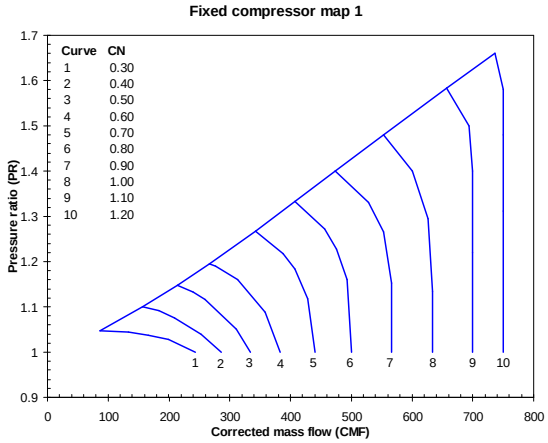


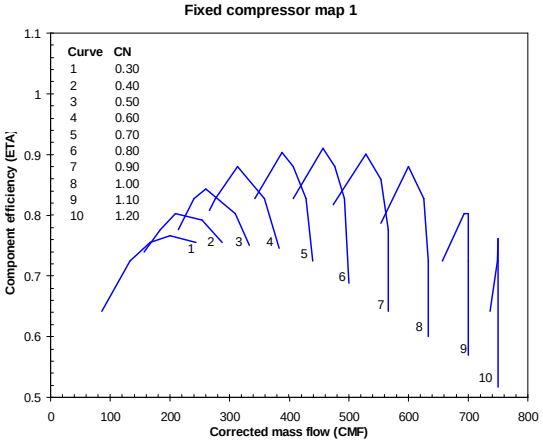


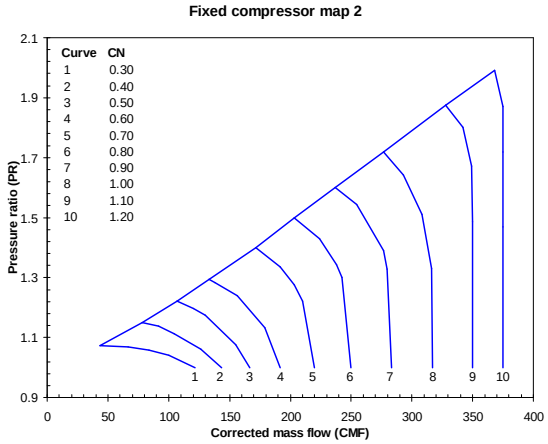


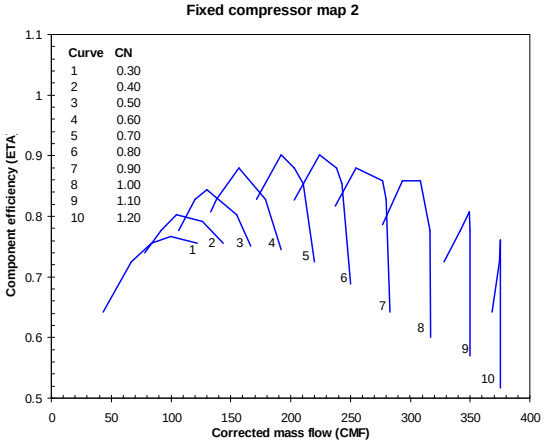


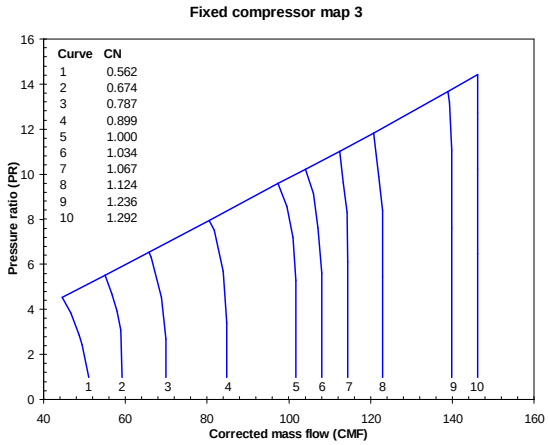
Appendix 2.7 Compressor Maps Used in Turbomatch

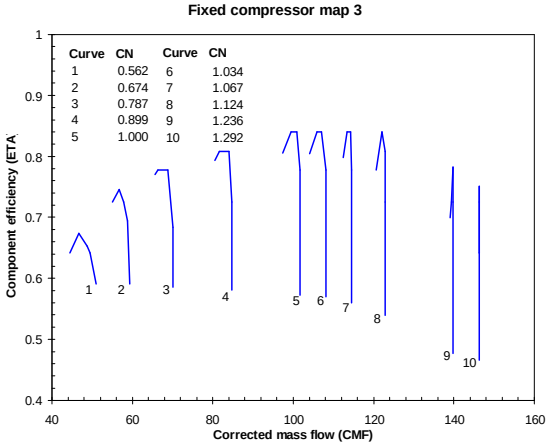


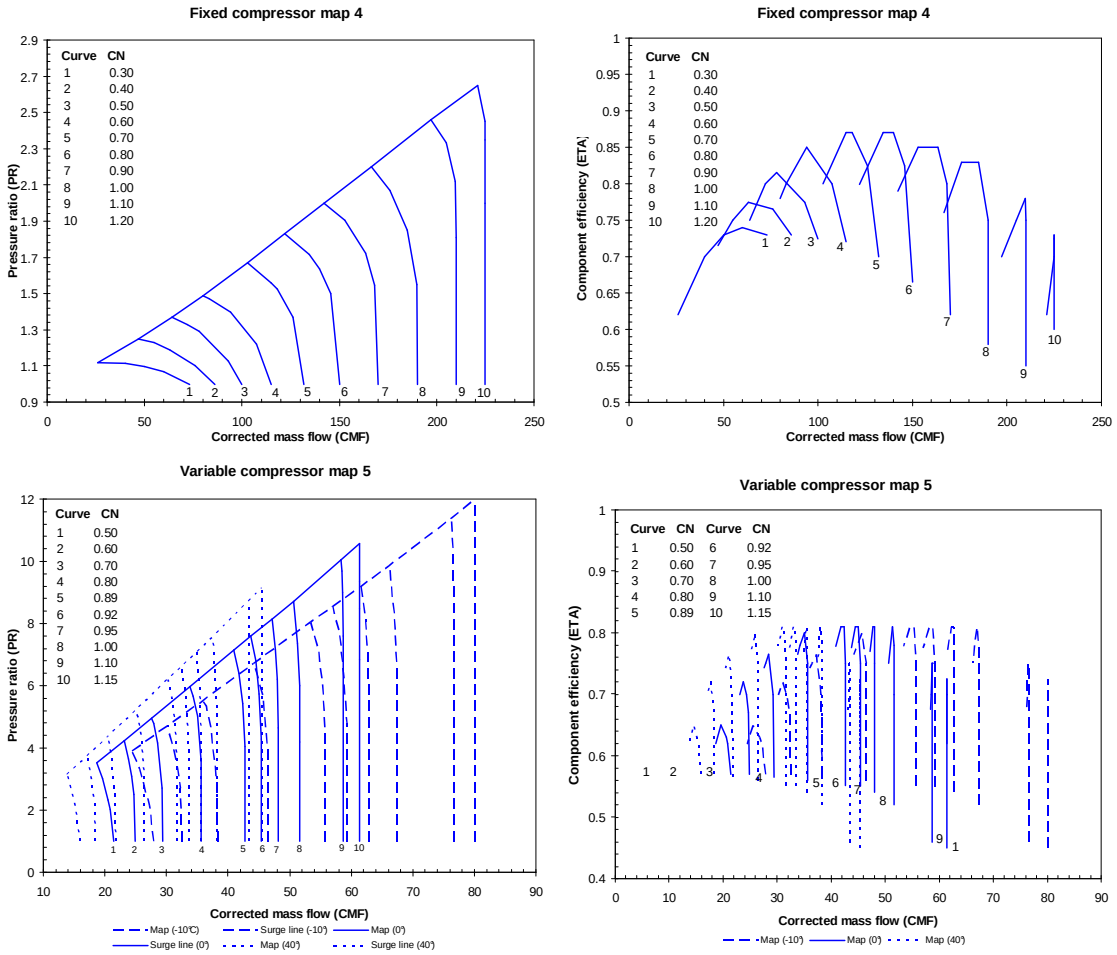






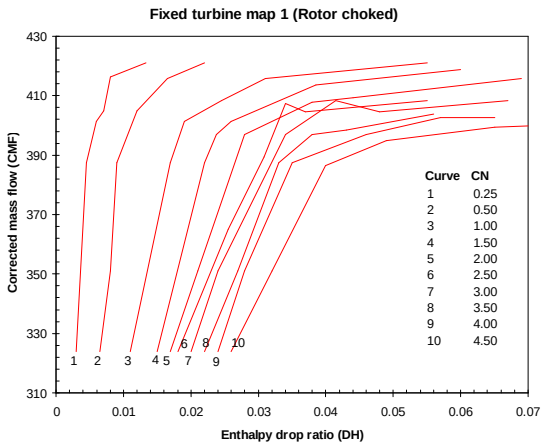


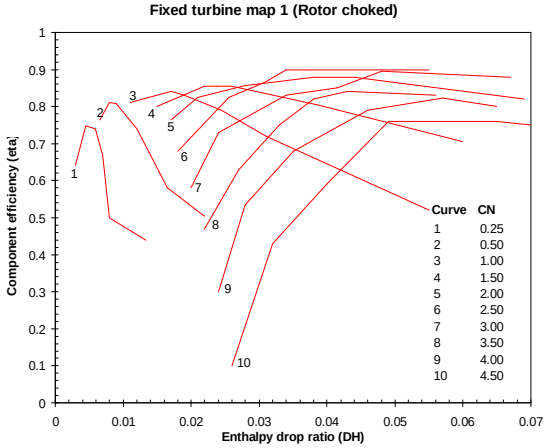


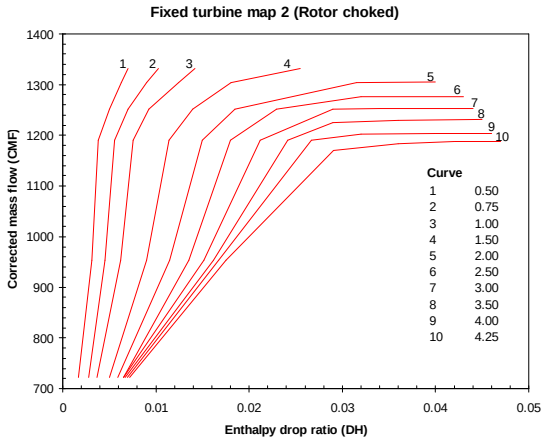


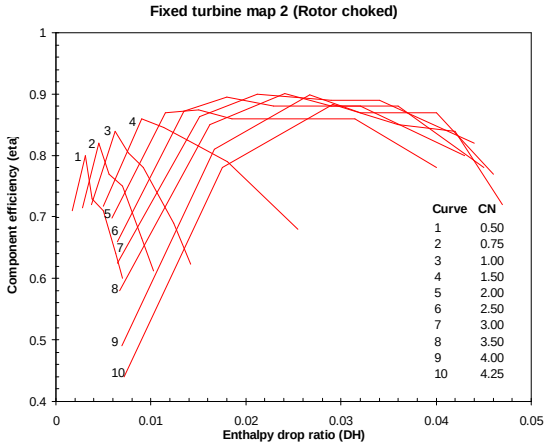
There are five different default compressor maps used in Turbomatch that are stored in map-folder specified in TURBOMATCHpreferences.txt file. The default maps have the ability to describe compressor performance with variable geometry. There is a different map for each VGV angle, as seen on last set of maps. VGV angles for which the charts are stored are -10°, 0°, 10°, 20°, 30° and 40°. If different angle is needed, corresponding map will be created by linear interpolation.

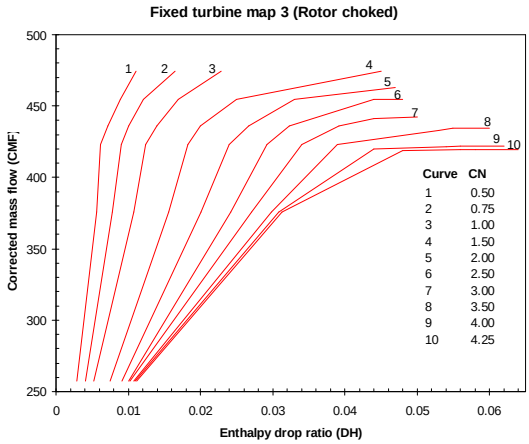
Appendix 2.8 Turbine Maps Used in Turbomatch

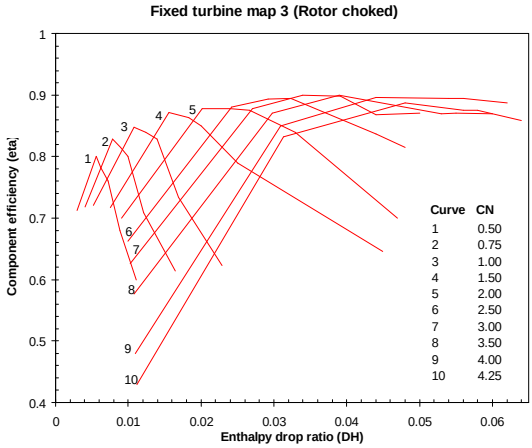


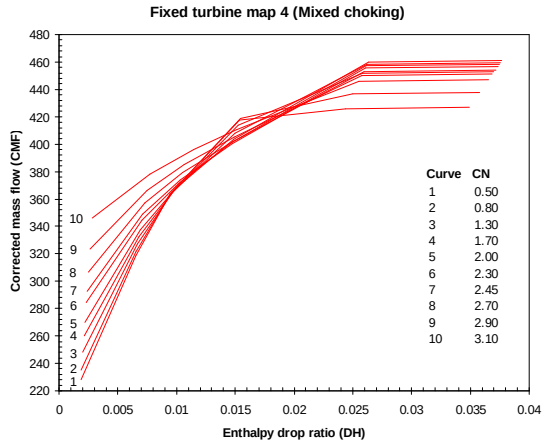


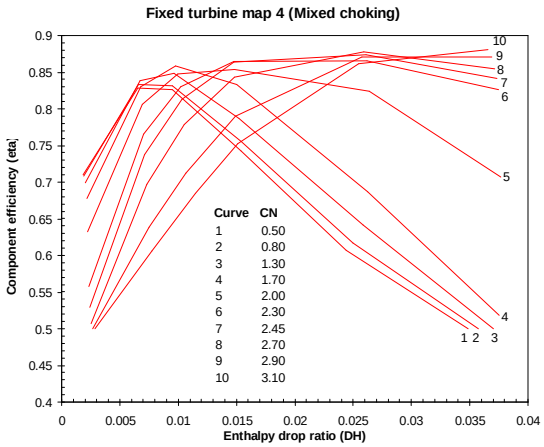


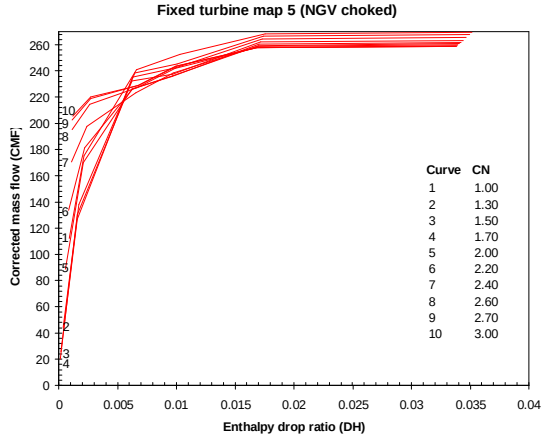


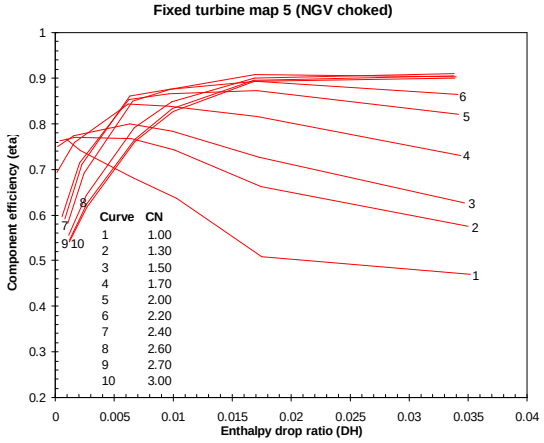


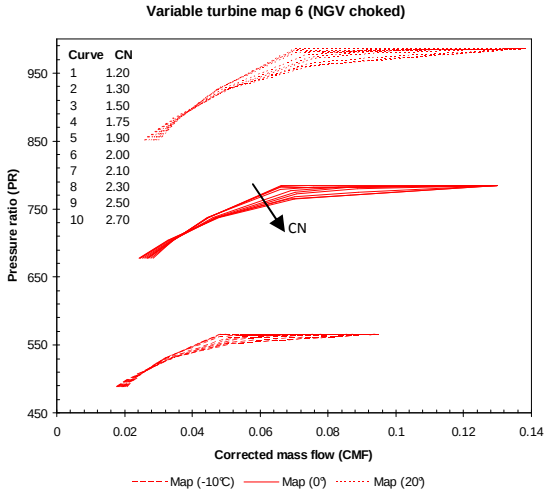


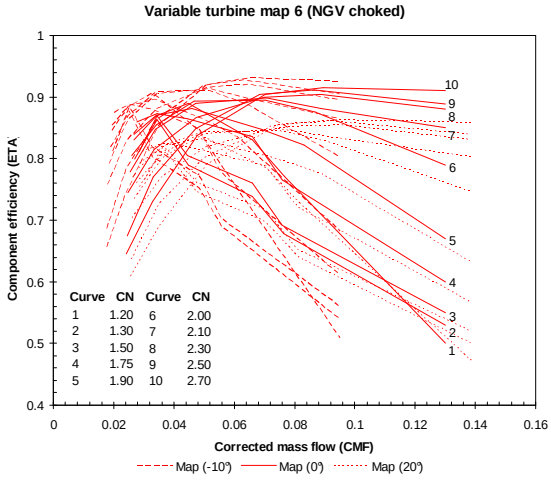












For the turbine there are six different default maps used in Turbomatch stored in the same folder as compressor maps. Every map has a unique shape which is primarily influenced by means of choking. First three maps correspond to a turbine where rotor chokes first, next map represents a map where either NGV or rotor are first choked, depending on operation conditions and the last two correspond to a turbine with NGV choking. NGV angles for which the charts are stored are -10°, 0°, 10° and 20°.

Appendix 2.9 Beta Lines

Compressor map is stored in three two-rank array variables in following manner:

Pressure ratio: piMap(CN, beta)

Corrected mass flow: CWmap(CN,beta)

Isentropic efficiency: etaCMap(CN,beta)

Corrected rotational speed values are stored in one-rank array: CNCMap(beta)

For better clarification following two tables demonstrates how rotational speed, pressure ratio and corrected mass flow are stored in Fortran array variables

Corrected rotational speed:

Declaration: real, dimension(:), allocatable :: CNCMap

Subscript	1	2	3	4	...
CN value	0.0651	0.1301	0.1951	0.2601	...

Compressor pressure ratio:

Declaration: real, dimension(:,:), allocatable :: piMap

	piMap(:,1)	piMap(:,2)	piMap(:,3)	piMap(:,4)	...
piMap(1,:)	1.0000	1.0000	1.0000	1.0000	...
piMap(2,:)	1.0012	1.0052	1.0118	1.0210	...
piMap(3,:)	1.0025	1.0104	1.0236	1.0420	...
piMap(4,:)	1.0037	1.0156	1.0353	1.0986	...
...	...	...	...	...	...

Corrected ratio:

Declaration: real, dimension(:,:), allocatable :: CWMap

	CWMap(:,1)	CWMap(:,2)	CWMap(:,3)	CWMap(:,4)	...
CWMap(1,:)	3.8004	7.3713	10.5597	13.7449	...
CWMap(2,:)	3.7999	7.3713	10.5597	13.7449	...
CWMap(3,:)	3.7993	7.3711	10.5595	13.7447	...
CWMap(4,:)	3.7980	7.3703	10.5588	13.7440	...
...	...	...	...	...	...

Isentropic efficiency map values are stored in a similar way as values of pressure ratio map. A point from the map is identified by two integer numbers enclosed in parentheses, the subscript, of which the first number represents the rotational speed (from the CNCMap) and the second the beta line. This map is scaled according to engine design point. Referring to compressor brick, the values of corrected rotational speed and isentropic efficiency are obtained from the map where pressure ratio and corrected mass flow are guessed. But if they were guessed directly, determining the corresponding subscript from the table would be more difficult. If for example one pressure ratio guess was 1.003 the corresponding rotational speed could be any number from the rotational speed array range. To determine corrected mass flow the computer will need to crate a new temporary array consisting of pairs of elements beta line points and rotational speed points corresponding to guessed pressure ratio. Some kind of pressure line will have to be created. The same would have to be done for corrected mass flow map where corrected mass flow line would be created. The final corresponding values of beta line and corrected rotational speed for pressure ratio and corrected mass flow guesses would be determined by intersection of these two created lines.

A simpler solution is to guess the beta line instead of pressure ratio. Corresponding corrected rotational speed could then be easily interpolated from CWMap. This approach has also been used in Simgaen. When both beta and CN values are known reading the values of pressure ratio and isentropic efficiency is straightforward.

A system of two equations with tow unknowns is solved by using a Newton-Raphson method. As the initial guess for these unknowns the Design-point values are used.

Simgaen was deliberately created just to simulate one type of engine and test the transient method implemented later into Turbomatch. Although the variables for single shaft turbojet in Simgaen and Turbomatch differ slightly (e.q. surge margin instead of beta lines is guessed in Turbomatch), the essence of off-design calculation remains identical. Since Turbomatch is capable of calculating arbitrary engine

configuration it has more variables used as guesses and more errors for iteration. Details can be found in Appendix 2.10.

Appendix 2.10 Turbomatch Variables and Errors

Being capable of calculating any possible gas turbine engine configuration, Turbomatch has dynamic unknowns and errors. Following list shows components and their unknowns required for off-design simulation if corresponding components are used.

VARIABLE NAME	VARIABLE DECLARED	NOTES
Compressor - COMPRE		
Surge margin	bricktype2(i)%Z	
Relative rotational speed	bricktype2(i)%PCN	If two compressors are placed on one shaft, this variable is selected for only one of compressors
Turbine - TURBIN		
Power output	bricktype4(i)%WorkPower	Applies for power turbine only (PCN variable has to be switched off)
Corrected mass flow	bricktype4(i)%TF	
Corrected rotational speed	bricktype4(i)%PCN	Applies for power turbine only (Power output variable has to be switched off)
Mass flow splitter - PREMAS		
Mass flow split ratio	bricktype7(i)%LambdaW	Used for bypass only

The number of variables used in the engine model depends on the model configuration. The number of variables is fixed and in order to find their solution set if non-linear equations that define the relation between variables. The number of the equations must conform to number of variables. Equations are transposed into functions having variables as arguments (directly or indirectly). Values of the functions are called errors. Roots of the variables are found when all the errors equal zero. Subroutine ErrProc is responsible for the error processing. It uses Newton-Raphson method to solve equations (functions) by means of calculating guesses for variables and processing deviations of the errors (function values). Following list shows all possible functions in Turbomatch. As compared to variables, these function are used if the component for which they apply is included in the model:

Compressor – COMPRE

thisCOMPRE%ER1=(WA1-A%W)/A%W

- This error is only used if it is manually selected in input file

WA1: Inlet compressor mass flow calculated from the previous component output data

A%W: Inlet compressor mass flow obtained from the compressor map according to guessed operation point

Turbine – TURBIN

thisTURBIN%ER1=(TF-TFF)/TF

TF: Turbine inlet corrected mass flow calculated from the previous component output data

TFF: Turbine inlet corrected mass flow variable (guessed by ErrProc subroutine)

thisTURBIN%ER2=(DELHAB1-DELHT)/DELHAB1

DELHAB1: Enthalpy difference calculated either from power balance between turbine and compressor(s) driven (for compressor turbine), or from inlet and outlet turbine pressure (for power turbine).

DELHT: Enthalpy difference obtained from turbine maps

Convergent nozzle – NOZCON:

thisNOZCON%ER1 = (PREQ - A%P) / PREQ

PREQ: Outlet total outlet pressure calculated from the nozzle area

A%P: Inlet total pressure calculated from the preceding bick

Mass flow mixing duct with total pressure change – MIXFUL

thisMIXFUL%ER1=(B%PS-A%PS)/B%PS

A%PS: Inlet static pressure

B%PS: Outlet static pressure

Convergent-divergent nozzle – NOZDIV

thisNOZDIV%ER1=(AreaOfThroat-thisNOZDIV%ThroatArea)/AreaOfThroat

AreaOfThroat: Nozzle throat area calculated from the nozzle mass flow

thisNOZDIV%ThroatArea: Nozzle throat area calculated from engine design point

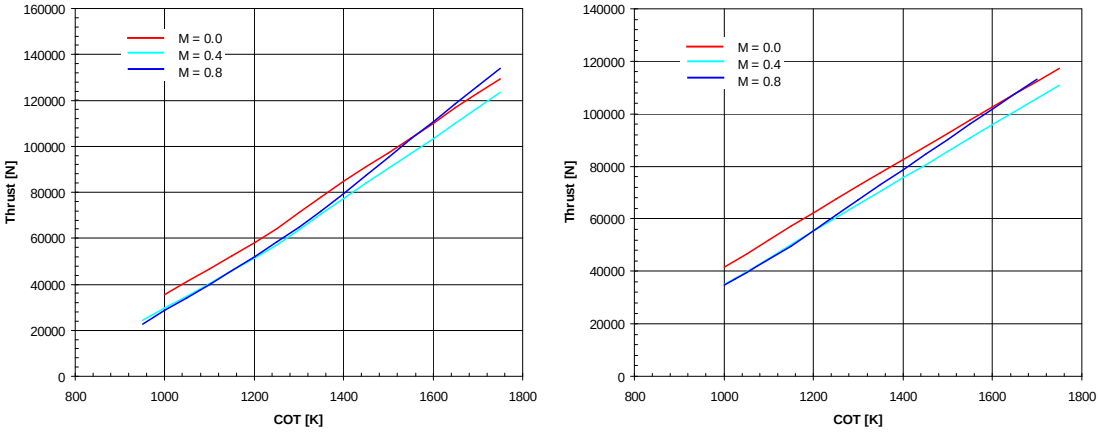
Appendix 2.11 Turbojet Model Design Point

A snippet of Turbomatch input file defining the engine configuration and operating design point is shown. This snippet is followed by a sequence of off-design operating point definition. Simgaen turbojet model posses the same configuration and specifications as Turbomatch model, but the operating point is slightly shifted on turbine map due to Simgaen different scaling routine as compared to Turbomatch.

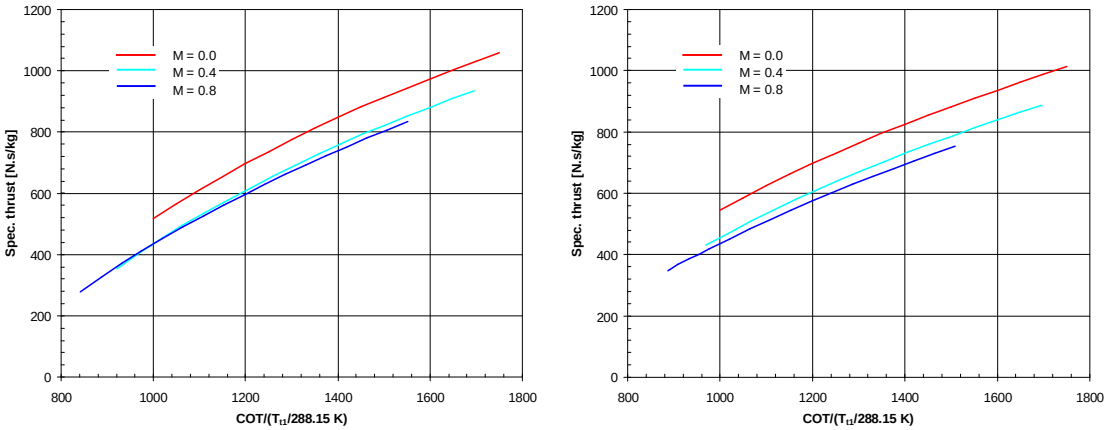
```
Turbojet example file
_____________________________________________________________
////
OD SI KE VA FP
-1
-1
INTAKE  S1,2       D1-6            R100
COMPRE  S2,3       D7-18           R102     V7    V8
BURNER  S3,4       D19-26          R104
TURBIN  S4,5       D27-41                   V28
NOZCON  S5,6,1     D42,43          R110
PERFOR  S1,0,0     D44-47,110,100,104,0,0,0,0,0
CODEND

DATA ITEMS ////
1 0.0            ! INTAKE: Altitude [m]
2 0.0              ! Deviation from ISA temperature [K]
3 0.0              ! Mach number
4 -1               ! Pressure recovery, according to USAF
5 0.               ! Deviation from ISA pressure [atm]
6 0.               ! Relative humidity [%]

7 0.75           ! COMPRESSOR I: Z = (R-R[choke])/(R[surge]-R[choke
8 1.               ! Relative rotational speed PCN
9 8.8              ! DP Pressure ratio
10 0.85            ! isentropic efficiency
11 0.              ! Error selection
12 5.              ! Compressor Map Number
13 1.              ! Shaft number
14 1.              ! Scaling factor of Pressure Ratio – Degradation factor
15 1.              ! Scaling factor of Non-D Mass Flow – Degradation factor
16 1.              ! Scaling factor of ETAc
17 0.1             ! Effective component volume [m^3]
18 0.              ! Stator angle (VSV) relative to DP

19 0.05          ! COMBUSTOR: Pressure
20 0.99            ! Combustion efficiency
21 -1              ! Fuel flow
22 0.              ! (>0) Water flow [kg s-1 or lb s-1] or (<0) Water to air ratio
23 288.            ! Temperature of water stream [K]
24 0.              ! Phase of water (0=liquid, 1=vapour)
25 1.              ! Scaling factor of ETAb (combustion efficiency
26 -1              ! Effective component volume [m^3]

27 0.            ! HP TURBINE: Auxiliary or power output [W]
28 0.6             ! Relative non-dimensional massflow W/
29 0.6             ! Relative non-rimensional speed CN (if = -1, value 0.6 is invoked)
30 0.86            ! Design isentropic efficiency
31 -1.             ! Relative non-dimensional speed PCN (= -1 for compressor turbine)
32 1.              ! Shaft Number (for power turbine, the value “0.” is used)
```

```
33 5                ! Turbine map umber
34 -1.              ! Power law index "n" (POWER = PCN^n) If = -1: power constant
35 1.               ! Scaling factor of TF (non-D inlet mass flow) – Degradation factor
36 1.               ! Scaling factor of DH (enthalpy change) – Degradation factor
37 1.               ! Scaling factor of ETAt is (Turbine isentropic efficiency
38 200.             ! Rotor rotational speed [RPS]
39 60.              ! Rotor moment of inertia [kg.m^2]
40 0.2              ! Effective component volume [m^3]
41 0.               ! NGV angle, relative to D.P.

42 -1.           ! CONVERGENT NOZZLE: Swich set (= "1" if exit area "floats"
                    !                          = "-1" if exit area is fixed)
43 1.               ! Scaling factor


44 -1.           ! ENGINE RESULTS: Power output - Power or Power
45 -1.              ! Propeller efficiency (= -1 for turbojet/turbofan)
46 0.               ! Scaling index ("1" = scalling needed, "0" = no scaling)
47 0.               ! Required DP net thrust or shaft power
                    ! = 0 if Scaling index = 0

-1
1 2 100.0           ! item 2 at station 1 = Mass flow(kg/s)
4 6 1400.0
-1
```

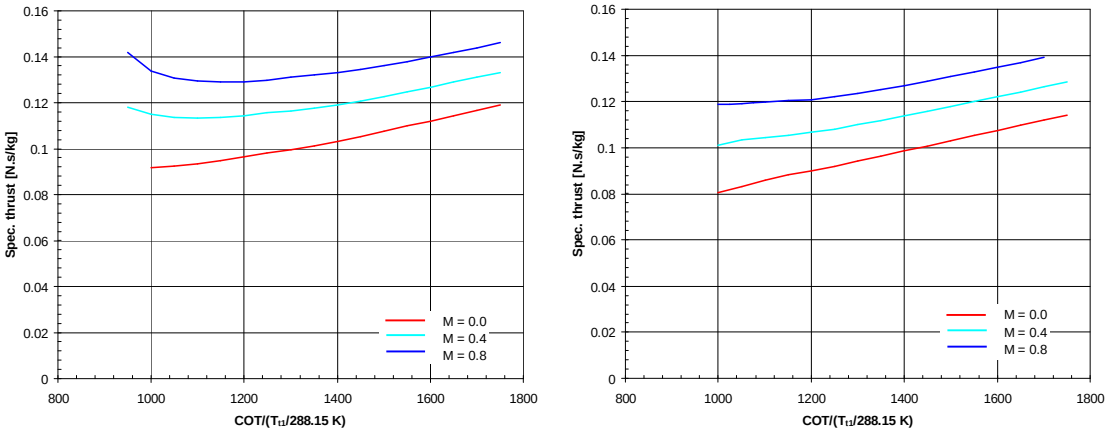
Appendix 2.12 The Comparison of Turbojet Off-design Model Results



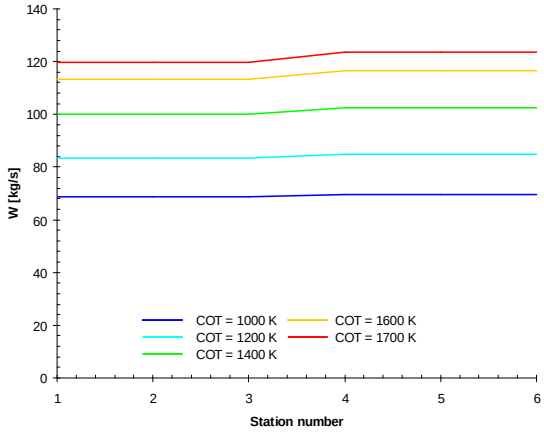
A 2.12.1 Engine net thrust evolution - Simgaen simulation (left), Turbojet simulation (right)

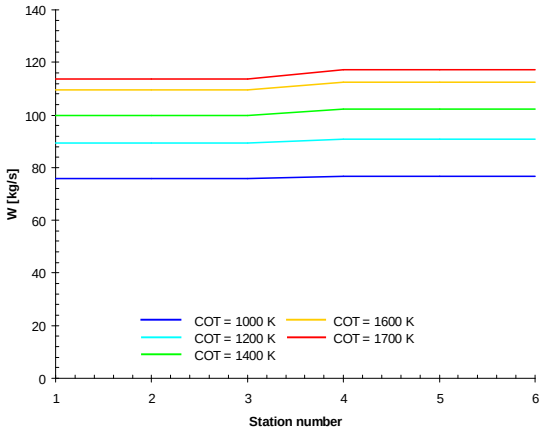


A 2.12.2 Engine specific thrust vs. corrected COT - Simgaen simulation (left), Turbojet simulation (right)

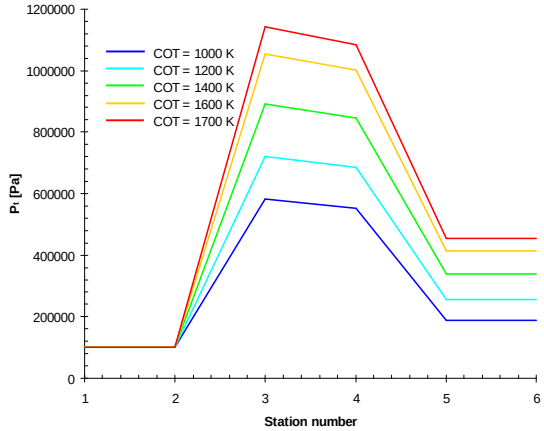


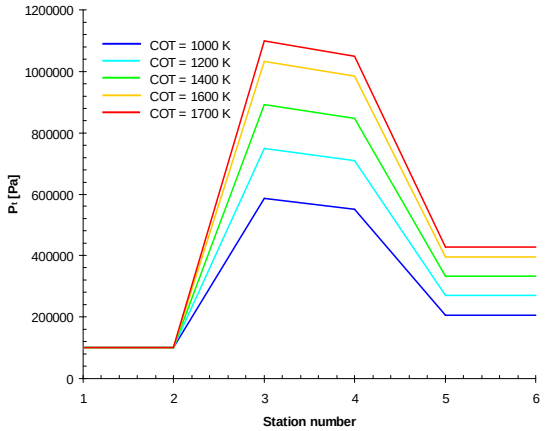
A 2.12.3 Thrust specific fuel consumption vs. corrected COT - Simgaen simulation (left), Turbojet simulation (right)



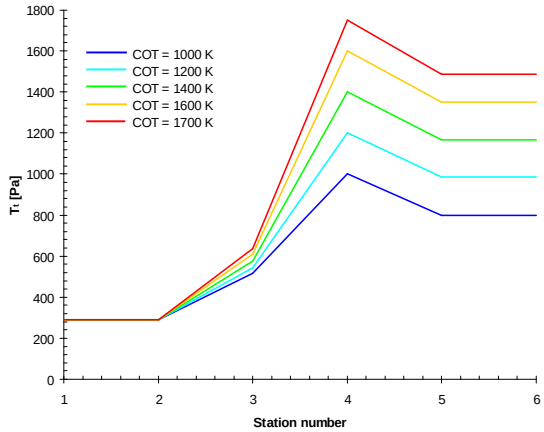


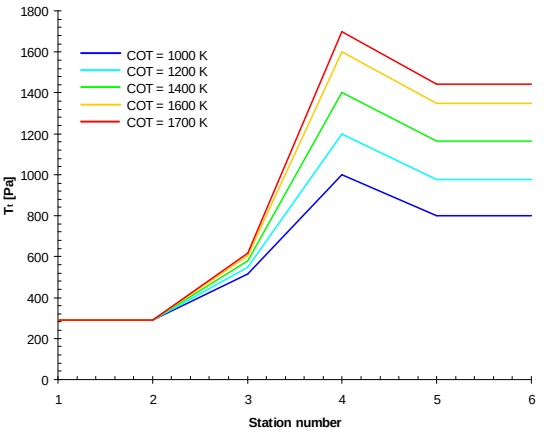
A 2.12.4 Mass flow evolution for M = 0.0 - Simgaen simulation (left), Turbojet simulation (right)





A 2.12.5 Total pressure evolution M = 0.0 - Simgaen simulation (left), Turbojet simulation (right)





A 2.12.6 Total temperature evolution M = 0.0 - Simgaen simulation (left), Turbojet simulation (right)

Appendix 2.13 Scaling Technique Comparison between Simgaen and Turbomatch

In both engine performance programs component maps are scaled to match the design point input by the user. For the compressor four data are input to describe the position of design point on the maps:

- Surge margin

$$Z = \frac{PR_{DP} - PR_{map\ \min}}{PR_{map\ surge} - PR_{map\ \min}}$$

where:

- all pressure ratios lie on the same speed line
- $PR_{map\ min} = 1.00$

NOTE: The definition of surge margin differs from the conventional surge margin definition. In the programs if the surge margin equals unity it belongs to a point located on the compressor map surge line. If this point surge margin was evaluate according to the conventional surge margin definition its value would equal to zero (Kurzke, Riegler 2000).

- Pressure ratio
- Relative rotational speed
- Isentropic efficiency.

The turbine design point on the maps is defined by:

- Relative non-dimensional mass flow
- Relative non-dimensional speed
- Isentropic efficiency
- Enthalpy difference that is calculated by work balance relationship according to the compressor work.

Because there are two maps available for every component only two parameters suffice to define the design point location. Parameter scaling ensures that component

maps correspond to design point determined by all input parameters. Correlations for scaling factors used in Simgaen and Turbomatch are consistent:

Compressor map scaling factors:

$$PRSF = \frac{PR_{required} - 1}{PR_{map\ original} - 1}$$

$$CMFSF = \frac{CMF_{caclulated}}{CMF_{map\ original}}$$

$$\eta_C SF = \frac{\eta_{C\ required}}{\eta_{C\ map\ original}}$$

Scaled compressor map (final map used in the engine model):

$$PR_{map\ scaled} = PRSF \cdot \left(PR_{map\ original} - 1\right) + 1$$

$$CMF_{map\ scaled} = CMFSF \cdot CMF_{map\ original}$$

$$\eta_{C\ map\ scaled} = \eta_C SF \cdot \eta_{C\ map\ original}$$

NOTE: Parameters labeled as *map original* are obtained from the compressor map using values of rotational speed and surge margin.

Turbine map scaling factors:

$$TFSF = \frac{TF_{caclulated}}{TF_{map\ original}}$$

$$DHSF = \frac{DH_{caclulated}}{DH_{map\ original}}$$

$$\eta_T SF = \frac{\eta_{T\ required}}{\eta_{T\ map\ original}}$$

Scaled turbine map:

$$TF_{map\ scaled} = TFSF \cdot TF_{map\ original}$$

$$DH_{map\ scaled} = DHSF \cdot DHF_{map\ original}$$

$$\eta_C F_{map\ scaled} = \eta_C SF \cdot \eta_C F_{map\ original}$$

NOTE: Parameters labeled as *map original* are obtained from the turbine map using values of turbine rotational speed and inlet turbine mass flow (corrected).

Inconsistency in TFSF in Simgaen and Turbomatch

Corrected inlet turbine mass flow $TF_{map\ original}$ is calculated by from

$$TF_{map\ original} = \left(TF_{map\ \max} - TF_{map\ \min}\right) \cdot TF_{relative} + TF_{map\ \min}$$

where the value of $TF_{relative}$ is specified by the user (e.g. in Turbomatch it is the second brick data in the TURBIN brick). Parameters $TF_{map\ max}$ and $TF_{map\ min}$ represent the mass flow maximum and minimum values used for design point turbine mass flow calculation. These values are in the programs not determined in the same way. In Turbomatch the $TF_{map\ max}$ and $TF_{map\ min}$ values are fixed for the map and they don't depend on design point rotational speed. In turbine map file named as "TURBINx.txt" where x denotes the map number their values are found in the third line. First number is the value of $TF_{map\ min}$ and the second number represents the difference of maximum and minimum mass flow ($TF_{map\ max}$ - $TF_{map\ min}$). In Simgaen the values of $TF_{map\ max}$ and $TF_{map\ min}$ correspond to the highest and lowest value of corrected mass flow, found on the design point rotational speed curve. Ultimately the $TF_{map\ original}$ for Simgaen and Turbomatch is different even if the $TF_{relative}$ is the same.

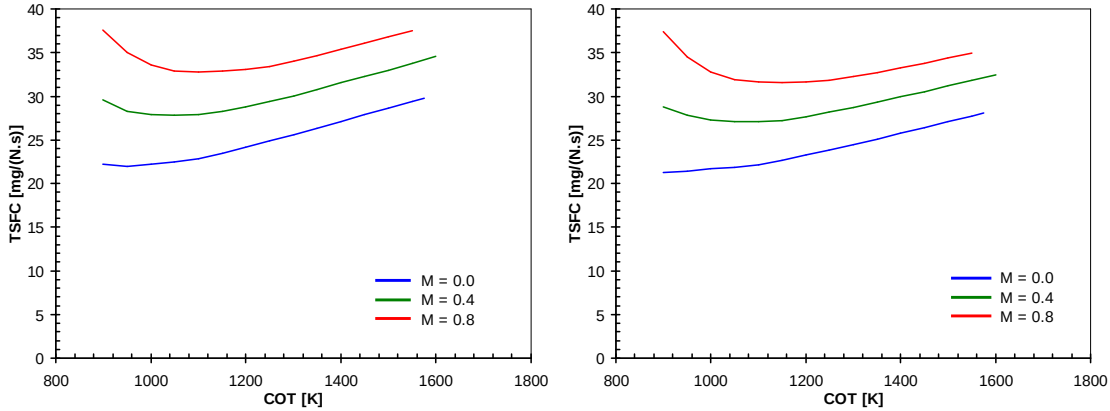
Appendix 2.14: RR Avon 300 Engine Input File for Turbomatch TR

The engine input file for Turbomatch Legacy ver. is found in public Turbomatch engine models library. This model has been adapted to conform the format of Turbomatch Intermediate and Transient version input file. The one for Transient version is shown here.

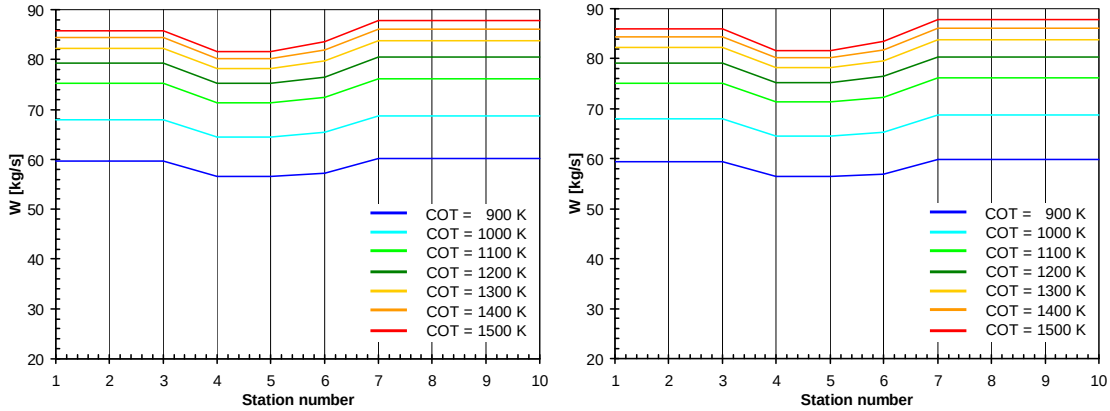
```
GT engine model for Turbomatch Transient ver.
Simulation of RR Avon 300
Ref: "Aircraft engines of the world 1960/61" by Paul H. Wilkinson, page 134
_____________________________________________________________
////
OD SI KE VA FP
-1
-1
INTAKE  S1,2       D1-6            R100
COMPRE  S2,3       D7-18           R102     V7    V8
PREMAS  S3,22,4    D19-22
DUCTER  S4,5       D23-27          R103
BURNER  S5,6       D28-35          R104
MIXEES  S6,22,7
TURBIN  S7,8       D36-50                   V37
DUCTER  S8,9       D51-55          R106
NOZCON  S9,10,1    D56,57          R110
PERFOR  S1,0,0     D58-61,110,100,104,0,0,0,0,0
CODEND

DATA ITEMS ////
1 0.              ! INTAKE: Altitude [m]
2 0.                 ! Deviation from ISA temperature [K]
3 0.                 ! Mach number
4 -1                 ! Pressure recovery, according to USAF
5 0.                 ! Deviation from ISA pressure [atm]
6 0.                 ! Relative humidity [%]

7 0.85            ! COMPRESSOR : Z = (R-R[choke])/(R[surge]-R[choke
8 1.                 ! Relative rotational speed PCN
9 8.8                ! DP Pressure ratio
10 0.84              ! isentropic efficiency
11 0.                ! Error selection
12 5.                ! Compressor Map Number
13 1.                ! Shaft number
14 1.                ! Scaling factor of Pressure Ratio – Degradation factor
15 1.                ! Scaling factor of Non-D Mass Flow – Degradation factor
16 1.                ! Scaling factor of ETAc is (Compressor isentropic efficiency)
17 0.1               ! Effective component volume [m^3]
18 0.                ! Stator angle (VSV) relative to DP

19 0.05            ! PREMAS: LAMDA W Cooling bypass (Wout/Win)
20 0.                ! DELTA W
21 1.                ! LAMBDA P
22 0.                ! DELTA P

23 0.             ! DUCTER: Air duct
24 0.001             ! Total pressure loss: DELTA(P)/Pin (Pressure loss = 1%)
25 0.                ! Combustion efficiency
26 0.                ! Limiting value of Fuel Flow (=100000 if not needed)
27 -1                ! Effective component volume [m^3]

28 0.05           ! COMBUSTOR: Pressure loss (=Total pressure loss/Inlet total pressure)
29 0.99              ! Combustion efficiency
30 -1                ! Fuel flow
31 0.                ! (>0) Water flow [kg s-1 or lb s-1] or (<0) Water to air ratio
32 288.              ! Temperature of water stream [K]
33 0.                ! Phase of water (0=liquid, 1=vapour)
```

```
34 1.               ! Scaling factor of ETAb (combustion efficiency)
35 0.02             ! Effective component volume [m^3]

36 0.            ! TURBINE: Auxiliary or power output [W]
37 0.8              ! Relative non-dimensional massflow W/Wmax
38 0.6              ! Relative non-rimensional speed CN (if = -1, value 0.6 is invoked)
39 0.87             ! Design isentropic efficiency
40 -1.              ! Relative non-dimensional speed PCN (= -1 for compressor turbine)
41 1.               ! Shaft Number (for power turbine, the value “0.” is used)
42 3.               ! Turbine map umber
43 -1.              ! Power law index "n" (POWER = PCN^n)
44 1.               ! Scaling factor of TF (non-D inlet mass flow) – Degradation factor
45 1.               ! Scaling factor of DH (enthalpy change) – Degradation factor
46 1.                  ! Scaling factor of ETAc is (Turbine isentropic efficiency) –
Degradation factor
47 200.             ! Rotor rotational speed [RPS]
48 50.              ! Rotor moment of inertia [kg.m^2]
49 0.2              ! Effective component volume [m^3]
50 0.               ! NGV angle, relative to D.P.

51 0.            ! DUCTER: Air duct
52 0.001            ! Total pressure loss: DELTA(P)/Pin (Pressure loss = 1%)
53 0.               ! Combustion efficiency
54 0.               ! Limiting value of Fuel Flow (=100000 if not needed)
55 -1               ! Effective component volume [m^3]

56 -1.           ! CONVERGENT NOZZLE: Swich set (= "1" if exit area "floats"
                    !                            = "-1" if exit area is fixed)
57 1.               ! Scaling factor


58 -1.              ! ENGINE RESULTS: Power output - Power or Power turbine (= -1 for
turbojet/turbofan)
59 -1.              ! Propeller efficiency (= -1 for turbojet/turbofan)
60 0.               ! Scaling index ("1" = scalling needed, "0" = no scaling)
61 0.                    ! Required DP net thrust(Turbojet,turbofan) or shaft power
(Turboprop.turboshaft)
                    ! = 0 if Scaling index = 0

-1
1 2 77.2            ! item 2 at station 1 = Mass flow (kg/s)
6 6 1141.0          ! item 6 at station 6 = Stagnation temperature (K)
-1
```

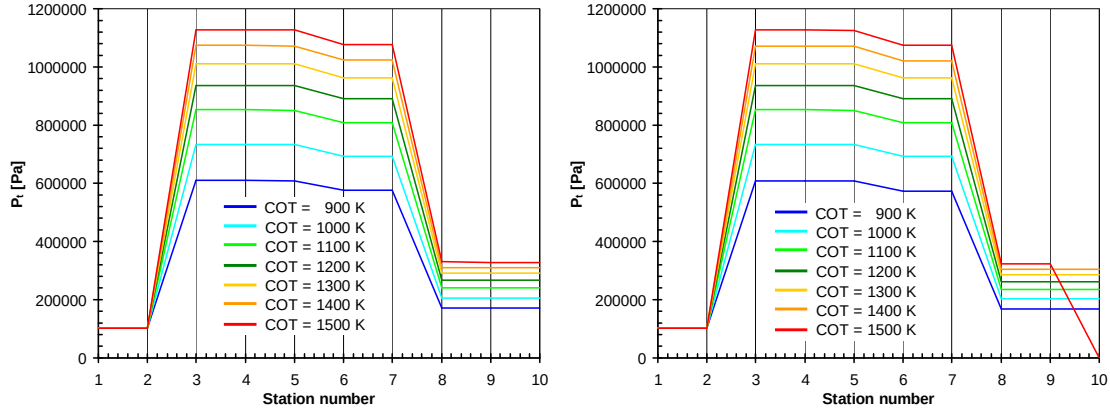
Appendix 2.15 RR Avon 300 Off-design Analysis in Turbomatch Legacy and TR version



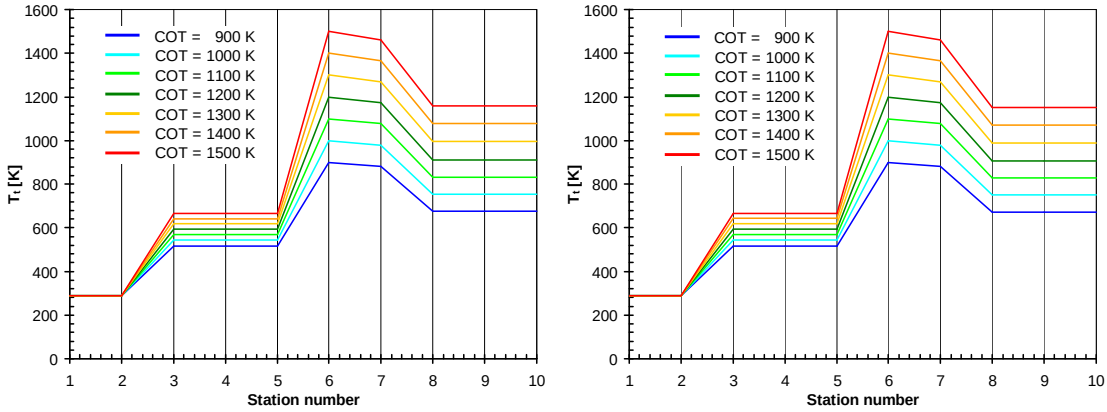
A2.15.1 Thrust specific fuel consumption variation (Sea level, dT_{ISA} = 0 °C) – Turbomatch Legacy version (left), Turbomatch Transient version (right)



A2.15.2 Mass flow evolution (Sea level static, dT_{ISA} = 0 °C) – Turbomatch Legacy version (left), Turbomatch Transient version (right)



A2.15.3 Total pressure evolution (Sea level static, dT_{ISA} = 0 °C) – Turbomatch Legacy version (left), Turbomatch Transient version (right)



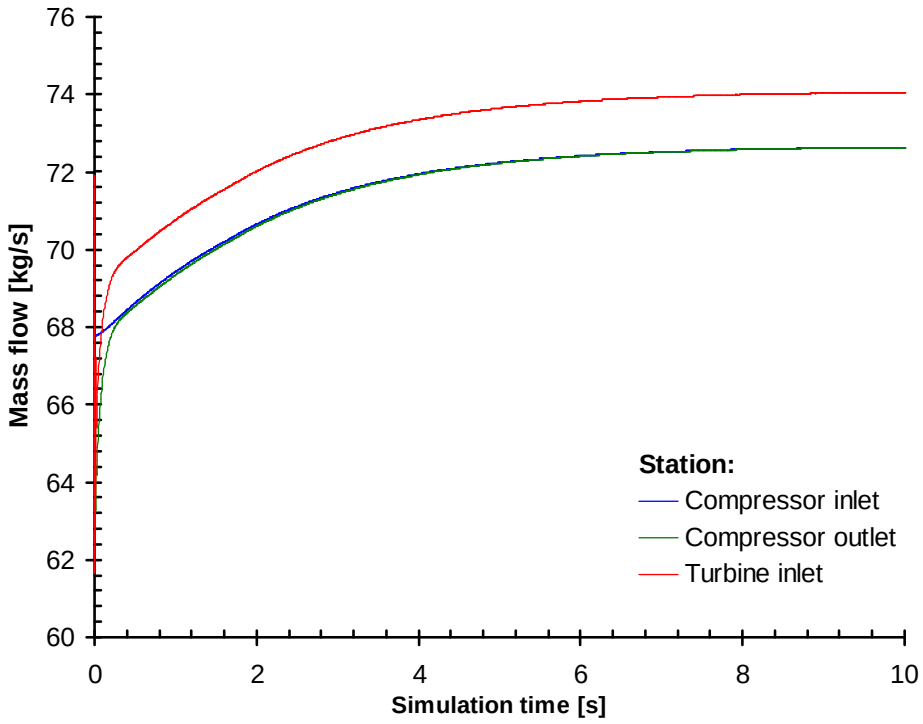
A2.15.3 Total temperature evolution (Sea level static, dT_{ISA} = 0 °C) – Turbomatch Legacy version (left), Turbomatch Transient version (right)

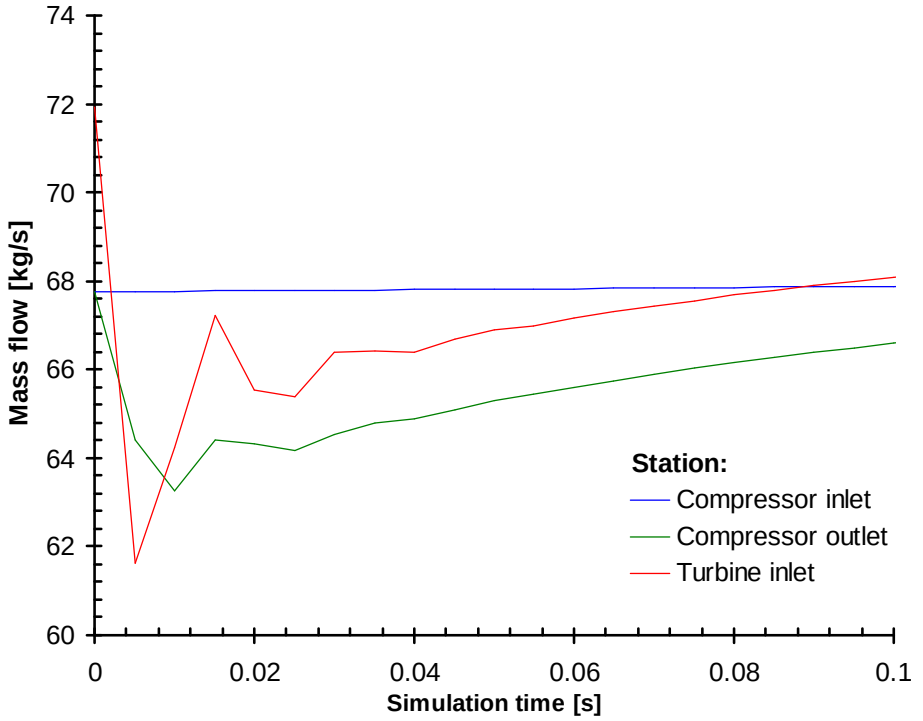
Appendix 2.16 Compressor Outlet Total Temperature Code Snippet

```
! Calculate phi PHIBS function from thermodynamic parameters:
PHIBS = TRM(A%F,A%T,3,A%WAR,A%X,A%P)&
        + GASCONST(A%F,A%WAR,A%X,A%P)*LOG(PR)

! Guess the value of outlet temperature for an isentropic process:
TBI=A%T*(PR**0.286_RK)
! Calculate outlet pressure:
B1%P=A%P*PR

! Evaluate outlet temperature for an isentropic process using phi
! function PHIBS in subroutine AIRTAB:
call AIRTAB(3,B1%F,TBI,PHIBS,B1%WAR,B1%X,B1%P,RETURN_STATUS)
! Check if successful:
if ( RETURN_STATUS/=0 ) then
   ! AIRTAB fails to converge on Temperature
   return
end if

ENTHA=TRM(A%F,A%T,2,A%WAR,A%X,A%P)
Calculate outlet enthalpy for a non-isentropic process:
ENTHB1=ENTHA+(TRM(B1%F,TBI,2,B1%WAR,B1%X,B1%P)-ENTHA)/ETA

! Guess the value of outlet temperature:
B1%T=A%T*(1.+(PR**0.286_RK-1._RK)/ETA)

! Evaluate outlet temperature using the value of outlet enthalpy in ! ! subroutine
AIRTAB:

AIRTAB(2,B1%F,B1%T,ENTHB1,B1%WAR,B1%X,B1%P,RETURN_STATUS)
! Check if successful:
if ( RETURN_STATUS/=0 ) then
   ! AIRTAB fails to converge on Temperature
   return
end if
```

Appendix 3.1 The Disturbances at Initial Steps of Transient Simulation

The mass flow variation during transient performance simulation of turbojet with a compressor volume 1.5 m^3 reveals disturbances at the initial stage of the simulation:

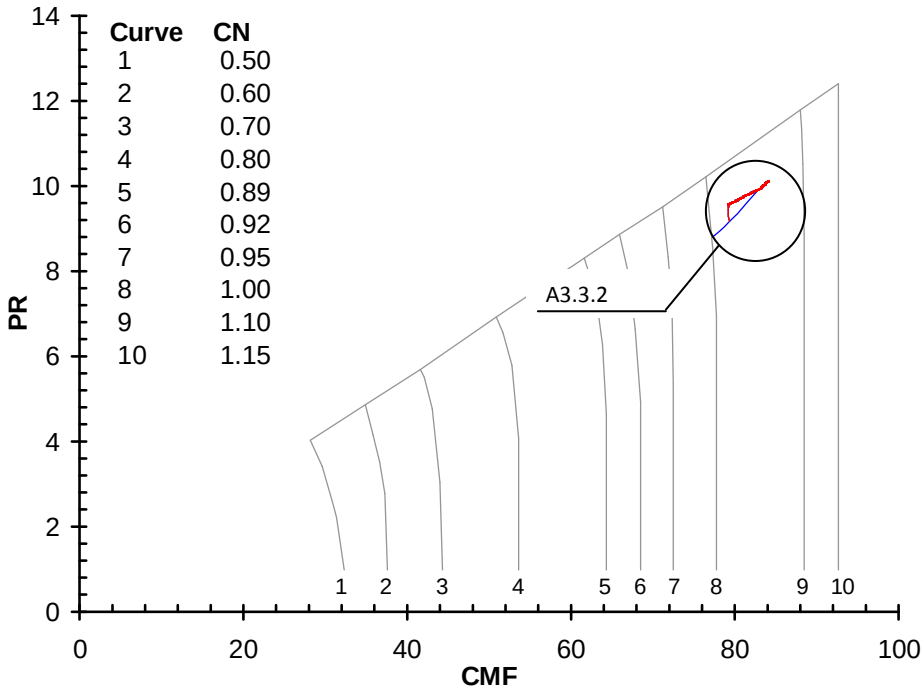




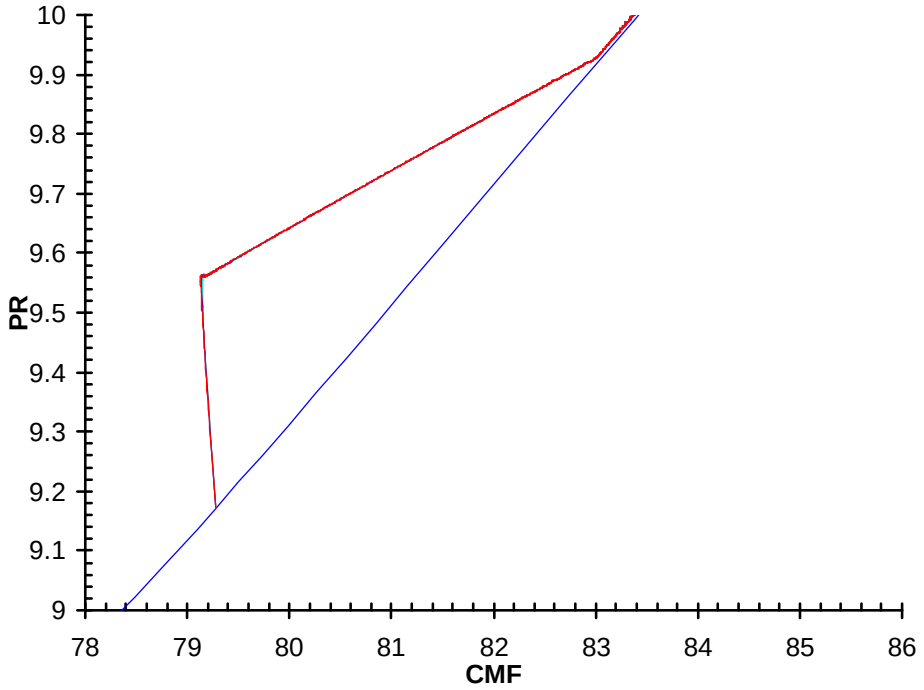
Appendix 3.2 NodeType variables

Variable	Description
BrickCount	The number of component in the input file count from the beginning
IBR	represents the brick ID (2=COMPRE, 3=BURNER,4=TURBIN,5=NOZCON,10=MIXFUL,12=NOZDIV, etc.)
NNUMB	number of the brick of the same type in the row
Inlet1Num, Inlet2Num	Station number of the first and second inlet respectively
Outlet1Num, Outlet2Num	Station number of the first and second outlet respectively
Prebrick, Postbrick	Points at preceeding and posterior FlowSegment according to the brick order specified in engine model input file
Inlet1, Inlet2	Points at the preceding brick which outlet represents the first and second inlet to this brick respectively
Outlet1, Outlet2	Points at the posterior brick which inlet represents the first and second outlet to this brick respectively
LambdaW	Component split ratio from E3.2.2
DeltaW	Component mass flow loss from E 3.2.2
k1, k2	Flow "k" coefficient
m1, m2	Flow "m" coefficient

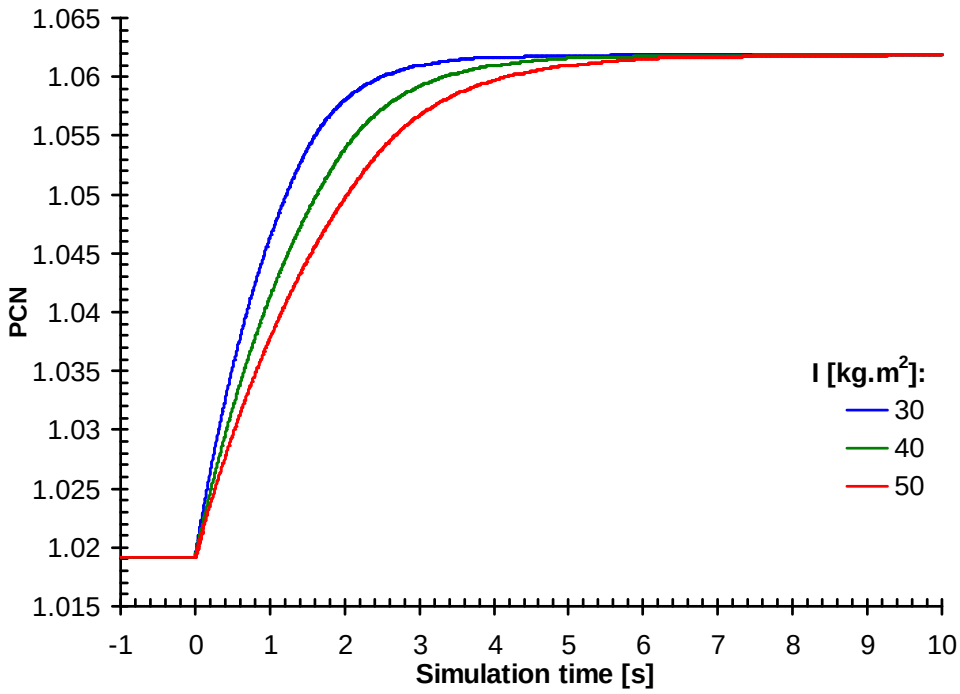
Appendix 3.3 Transient Performance of One Spool Gas Generator with Power Turbine



A3.3.1 Transient working line printed on compressor map

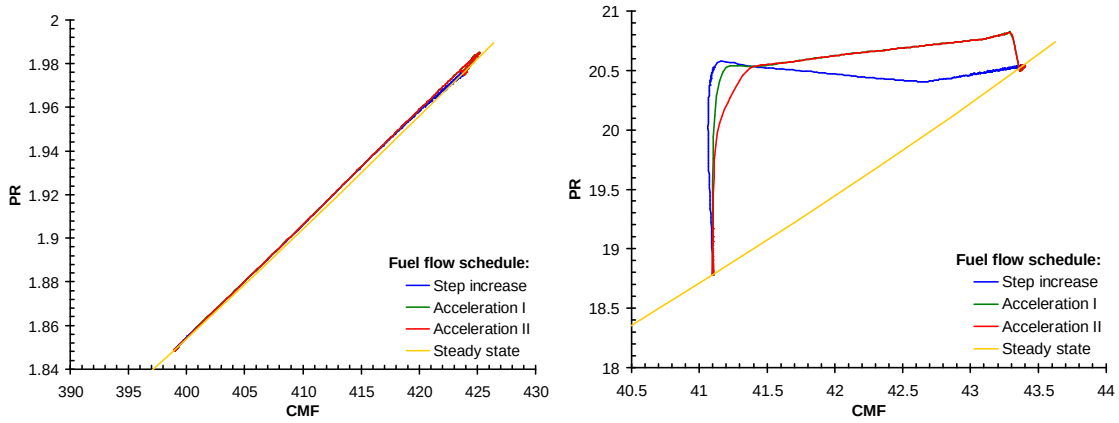


A3.3.2: The detail of compressor working line during transient performance. The kink point at the last part of transients is caused by the roughness of the turbine map

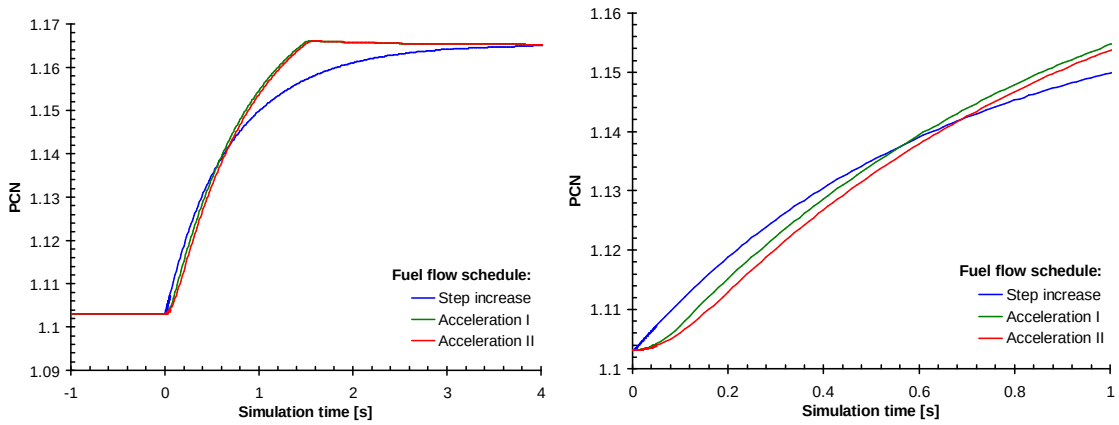


A3.3.3 The variation of compressor pressure ratio during the transient performance against the time

Appendix 3.4 Transient Performance of a Two-spool Turbofan

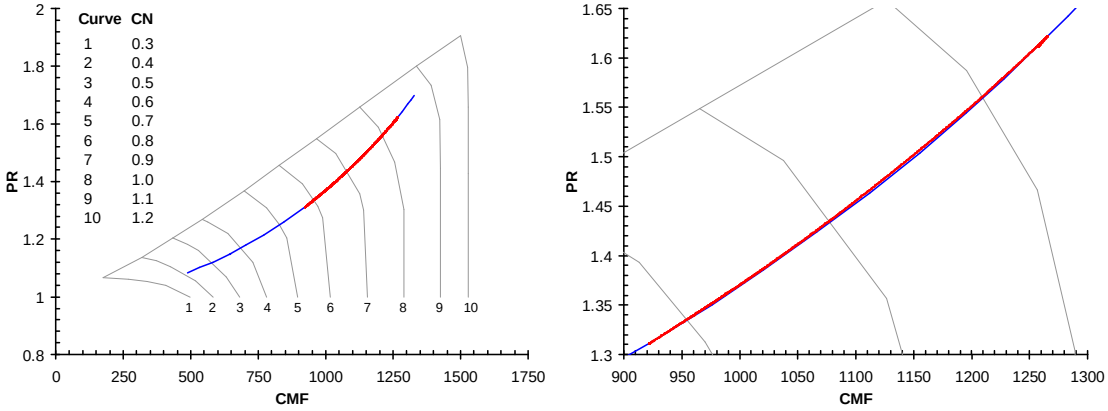


A3.4.1: The steady state and transient working line of the fan (left) and HP compressor (right)

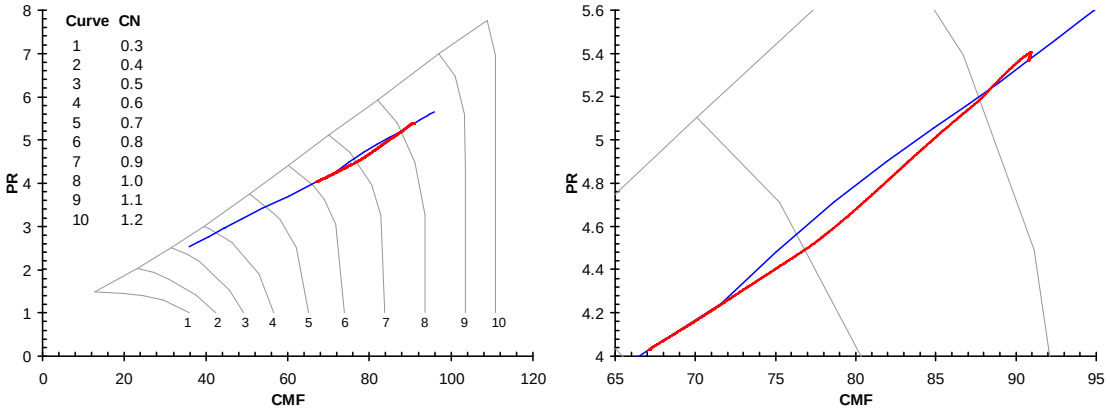


A3.4.2: The variation of the HP spool relative rotational speed during transient performance. Left image shows the variation during the first second of transients.

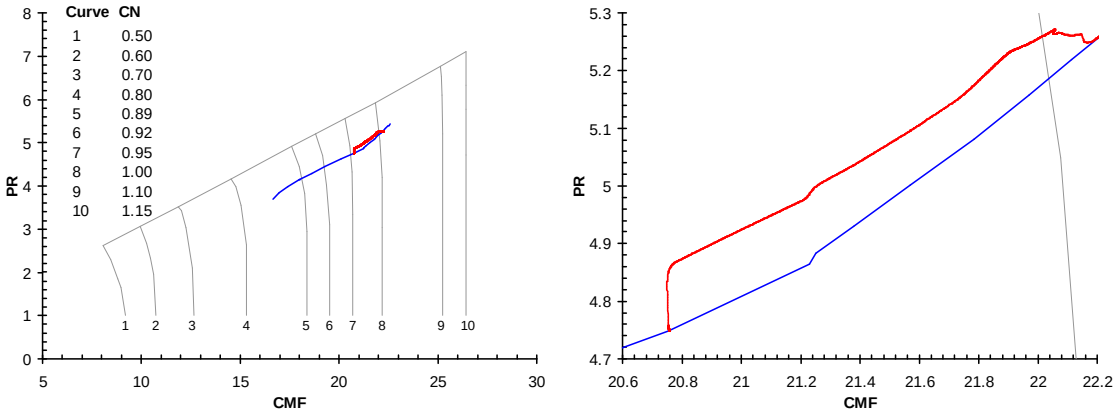
Appendix 3.5 Transient Performance of a Three-spool Turbofan



A3.5.1 Steady state (blue) and transient (red) LP compressor working line

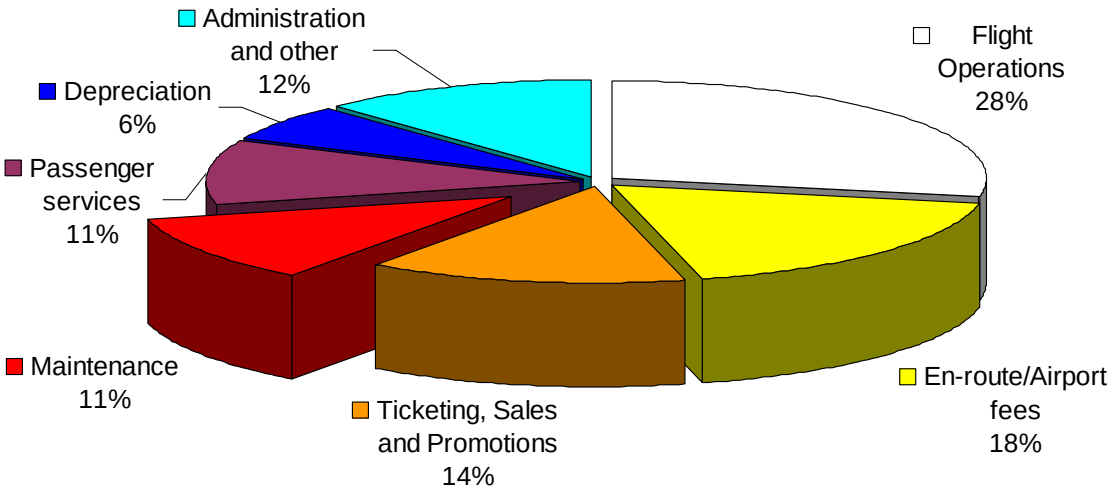


A3.5.2 Steady state (blue) and transient (red) IP compressor working line



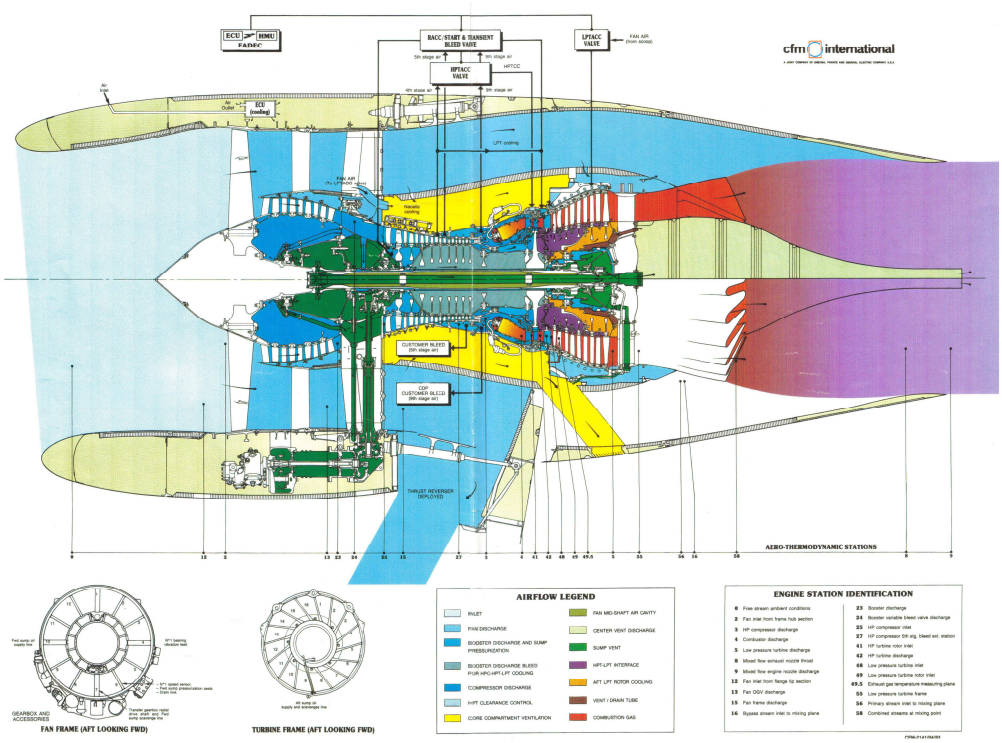
A3.5.3 Steady state (blue) and transient (red) HP compressor working line

Appendix 5.1



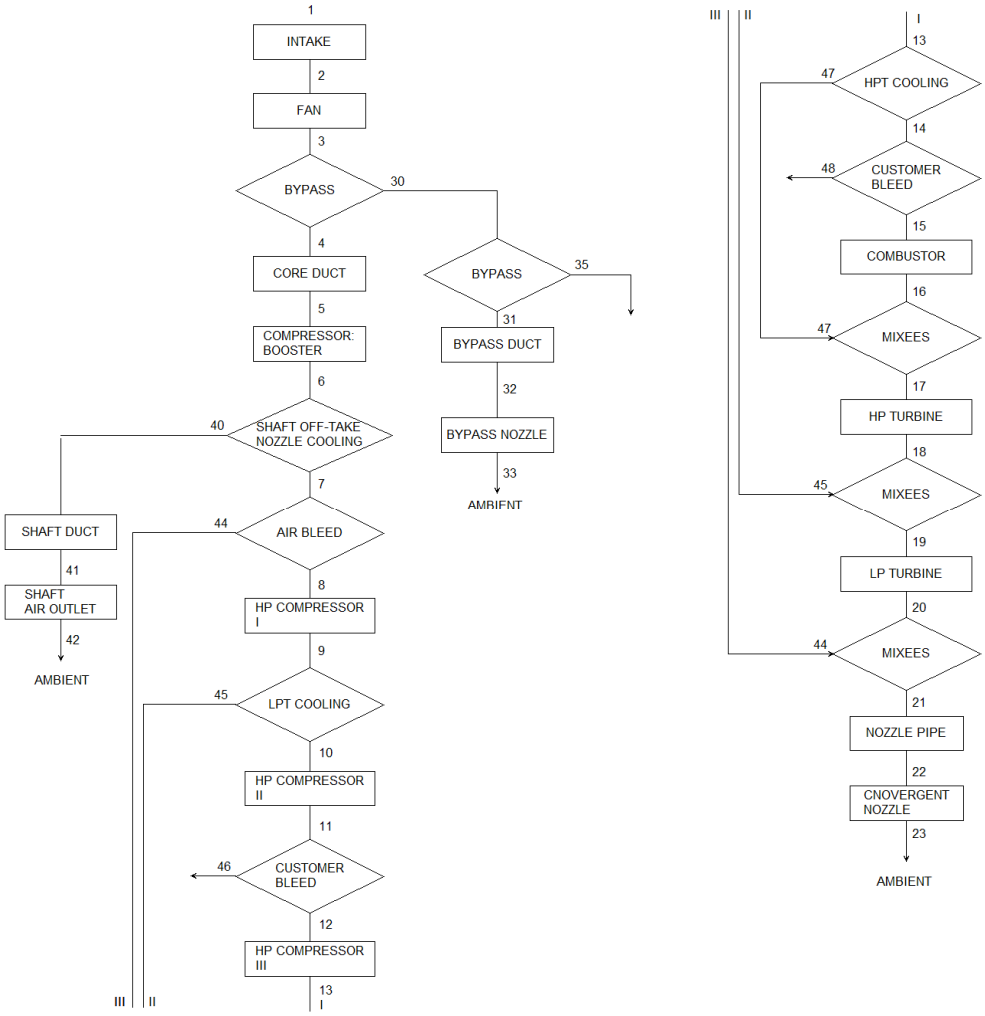
Aircraft operating costs breakdown (Source: ICAO world average 2001)

Appendix 5.2 Engine Mass Flow Diagram



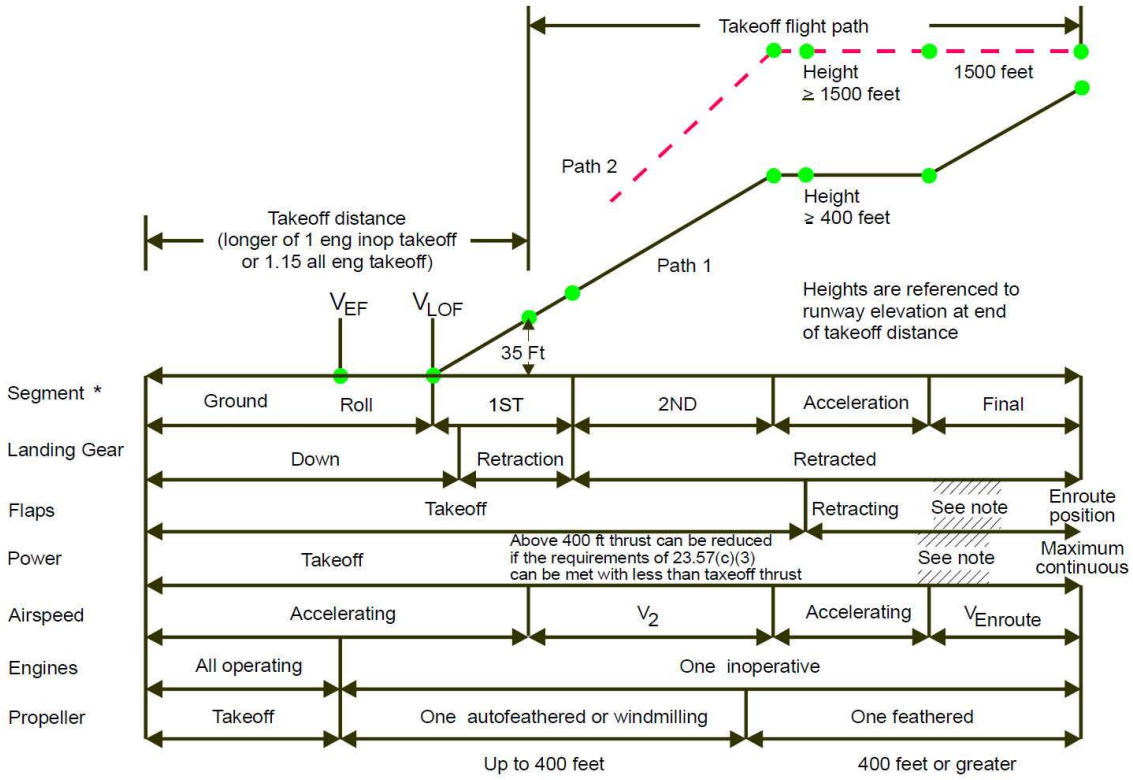
Source: CFM international

Appendix 5.3 Flow Chart of an Engine Model Used for Short Haul Flight Analysis



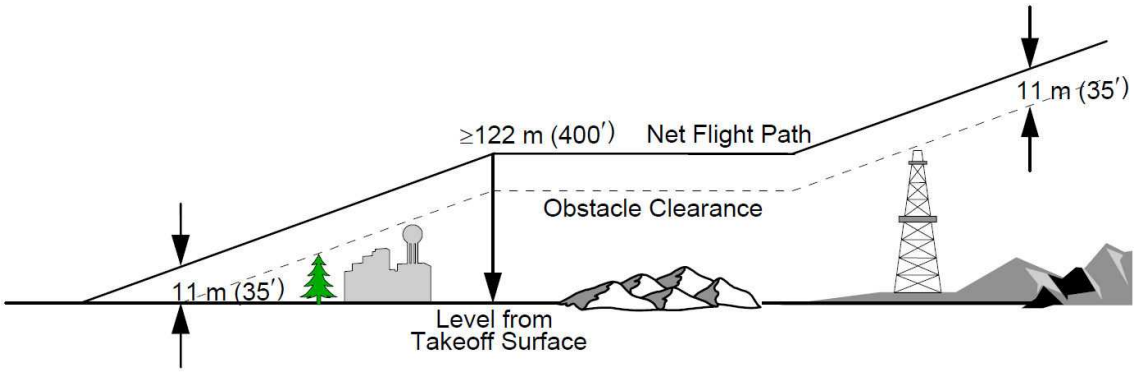
Appendix 5.4

The diagram of take-off segments and obstacle clearance approved by CS 25 decision(Goudou 2003a). Images were extracted from CS 23 (Goudou 2003b). Take-off segments recognized in the diagram correspond to take-off segments of Federal Aviation Regulations.



NOTE: The en route takeoff segment usually begins with the aeroplane in the en route configuration and with maximum continuous thrust, but it is not required that these conditions exist until the end of the takeoff path when compliance with 23.67(c)(3) is shown. The time limit on takeoff thrust cannot be exceeded.

A5.3.1 CS 23 approved take-off segments



A5.3.2 CS 23 obstacle clearance

Appendix 6.1 The Turbomatch Model of ICR Engine

```
Marine gas turbine with recuperated cycle
Date: 16.6.2009
_____________________________________________________________________________
////
TR SI KE VA FP
-1
-1
INTAKE  S1,2      D1-6            R300
COMPRE  S2,3      D7-18           R305     V7     V8
COOLER  S3,4      D19-29
COMPRE  S4,5      D30-41          R310     V30    V31
PREMAS  S5,30,6   D42-45
PREMAS  S30,31,32 D46-49
BURNER  S6,8      D55-62          R315
MIXEES  S8,31,9
TURBIN  S9-10     D63-77                   V64
MIXEES  S10,32,11
TURBIN  S11,12    D78-92                   V79
DUCTER  S12,13    D93-97          R320
TURBIN  S13,14    D98-112                  V98    V99
DUCTER  S14,16    D119-123        R325
NOZCON  S16,17,1  D124,125        R330
PERFOR  S1,0,0    D98,126-128,330,300,315,0,0,0,0,0,0
CODEND

DATA ITEMS////
1 0.            ! INTAKE: Altitude
2 0.              ! Deviation from ISA temperature
3 0.              ! Mach number
4 -1              ! Pressure recovery, according to USAF
5 0.              ! Deviation from ISA pressure [atm]
6 0.              ! Relative humidity [%]

7 0.80          ! LP COMPRESSOR: Z = (R-R[choke])/(R[surge]-R[choke])
8 1.0             ! DP Relative rotational speed PCN
9 3.01            ! DP Pressure ratio
10 0.86           ! DP isentropic efficiency
11 0.             ! Error selection
12 4.             ! Compressor Map Number
13 1.             ! Shaft number
14 1.             ! Scaling Factor of Pressure ratio
15 1.             ! Scaling Factor of Non-D Mass Flow
16 1.             ! Scaling Factor of Isentropic Efficiency
17 -1             ! Effective component volume [m^3]
18 0.             ! Stator angle (VSV) relative to DP

19 1             ! INTERCOOLER: ! Type of the
20 2              ! Medium used for cooling
21 0.78           ! Effectiveness [ dimensionless ]
22 0.03           ! Pressure loss
23 0.3            ! Design point Mach number of the working gas flow
24 50.            ! Cooling medium inlet mass flow [kg/s, lb/s]
25 1.             ! Cooling medium inlet total pressure [atm]
26 288.15         ! Cooling medium inlet total temperature  [K]
27 20.            ! Velocity of the cooling medium [m/s]
28 0.001          ! Average wall thickness [m]
29 2000.          ! Aspect ratio of the hot passage AR

30 0.80         ! HP COMPRESSOR: Z = (R-R[choke])/(R[surge]-R[choke])
31 1.             ! DP Relative rotational speed PCN
32 4.90           ! DP Pressure ratio
33 0.95           ! DP isentropic efficiency
34 1.             ! Error selection
35 4.             ! Compressor Map Number
36 2.             ! Shaft number
37 1.             ! Scaling Factor of Pressure Ratio
38 1.             ! Scaling Factor of Non-D Mass Flow
39 1.             ! Scaling Factor of Isentropic Efficiency
40 -1             ! Effective component volume [m^3]
```

```
41 0.              ! Stator angle (VSV) relative to DP

42 0.10          ! OVERALL COOLING - PREMAS: LAMDA W
43 0.              ! DELTA W
44 1.              ! LAMBDA P
45 0.              ! DELTA P

46 0.7           ! LPT COOLING (FROM OVERALL COOLING) - PREMAS: LAMDA W
47 0.              ! DELTA W
48 1.              ! LAMBDA P
49 0.              ! DELTA P

55 0.065         ! COMBUSTOR: Pressure loss
56 0.998           ! Combustion efficiency
57 -1              ! Fuel flow
58 0.              ! (>0) Water flow or (<0) Water to air ratio
59 288.            ! Temperature of water stream [K]
60 0.              ! Phase of water (0=liquid, 1=vapour)
61 1.              ! Scaling factor of ETAb (combustion efficiency)
62 -1              ! Effective component volume [m^3]

63 0.            ! HP TURBINE: Auxiliary or power output [W]
64 -1              ! DP Relative non-dimensional massflow W/Wmax
65 -1              ! DP Relative non-rimensional speed CN
66 0.87            ! DP isentropic efficiency
67 -1.             ! DP Relative non-dimensional speed
68 2.              ! Shaft Number
69 5.              ! Turbine map number
70 -1.             ! Power law index "n" (POWER = PCN^n)
71 1.              ! Scaling factor of TF (non-D inlet mass flow)
72 1.              ! Scaling factor of DH (enthalpy change)
73 1.              ! Scaling factor of ETAc is
74 160.            ! DP Rotor rotational speed [RPS]
75 20.             ! Rotor moment of inertia [kg.m^2]
76 -1              ! Effective component volume [m^3]
77 0.              ! NGV angle, relative to D.P.

78 0.            ! LP TURBINE: Auxiliary or power output [W]
79 0.8             ! DP Relative non-dimensional massflow W/Wmax
80 0.6             ! DP Relative non-rimensional speed CN
81 0.87            ! DP isentropic efficiency
82 -1.             ! DP Relative non-dimensional speed PCN
83 1.              ! Shaft Number
84 5.              ! Turbine map number
85 -1.             ! Power law index "n" (POWER = PCN^n)
86 1.              ! Scaling factor of TF (non-D inlet mass flow)
87 1.              ! Scaling factor of DH (enthalpy change)
88 1.              ! Scaling factor of ETAc is
89 120.            ! DP Rotor rotational speed [RPS]
90 30.             ! Rotor moment of inertia [kg.m^2]
91 -1              ! Effective component volume [m^3]
92 0.              ! NGV angle, relative to D.P.

93 0.            ! TURBINE DUCT - DUCTER: Air duct
94 0.005           ! Total pressure loss: DELTA(P)/Pin
95 0.              ! Combustion efficiency
96 0.              ! Limiting value of Fuel Flow
97 -1              ! Effective component volume [m^3]

98 -1.15         ! POWER TURBINE: negative value of exit pressure
99 -1              ! DP Relative non-dimensional massflow W/Wmax
100 0.8            ! DP Relative non-rimensional speed CN
101 0.89           ! DP isentropic efficiency
102 0.9            ! DP Relative non-dimensional speed PCN
103 0.             ! Shaft Number
104 5.             ! Turbine map number
105 -1.            ! Power law index "n" (POWER = PCN^n)
106 1.             ! Scaling factor of TF (non-D inlet mass flow)
107 1.             ! Scaling factor of DH (enthalpy change)
108 1.             ! Scaling factor of ETAc is
109 50.            ! DP Rotor rotational speed [RPS]
110 30.            ! Rotor moment of inertia [kg.m^2]
111 -1             ! Effective component volume [m^3]
```

```
112 0.             ! NGV angle, relative to D.P.

119 0.           ! EXHAUST DUCT - DUCTER: Air duct
120 0.005           ! Total pressure loss: DELTA(P)/Pin
121 0.              ! Combustion efficiency
122 0.              ! Limiting value of Fuel Flow
123 -1              ! Effective component volume [m^3]

124 -1.          ! NOZZLE: Swich set
125 1.              ! Scaling factor

126 -1.          ! ENGINE RESULTS - PERFOR: Propeller efficiency
127 0.              ! Scaling index
128 0.              ! Required DP net thrust or shaft power
                    ! = 0 if Scaling index = 0

-1
1 2  72.5           ! Inlet mass flow
4 6  310.00         ! Intercooler outlet temperature
8 6 1455
-1
```